LEETECH

LeeTech DBA/Query

Terminal Mode

LeeTech Software Inc.

Nobody provides better service than we do.

LeeTech DBA/Query

Terminal Mode Reference Manual

LeeTech
SOFTWARE INC.

2085 Hamilton Ave., Suite 200, San Jose, CA 95125

Released 01/03/01

The information contained in this document is subject to change without notice.

LEETECH SOFTWARE INC. MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. LeeTech Software Inc. shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance or use of this material.

TABLE OF CONTENTS

PART 2

FIGURES

TABLES

1. Introduction

DBA/Query is a data manipulator, an ad hoc report generator and a database real-time monitor for ALLBASE/SQL.

User Interface

DBA/Query provides ALLBASE/SQL users with an interface similar to that of TurboIMAGE QUERY.PUB.SYS. An experienced TurboIMAGE user can easily master ALLBASE/SQL (making it do FORM, FIND, ADD, UPDATE, DELETE and REPORT) in no time. Most of the TurboIMAGE report procedures can be used on ALLBASE/SQL databases without any changes.

Real-time Monitor

The database real-time monitor diagnoses ALLBASE/SQL runtime behaviors and analyzes performance problems. There is no longer a need to execute Update Statistics for database capacity analysis, because the powerful monitor DBA/Query presents real-time DBEFileSet utilization reports in just a few seconds while other users are concurrently accessing the database. In a report summary, DBA/Query can display data regarding different MPE/XL users, application programs running, database internal functions and running state. This can all be utilized to evaluate LOCK, LATCH, performance or system hanging problems. DBA/Query calculates tables, views row counts in real-time, shows runtime control block/buffer size, displays, global counter usage, examines logfile utilization and analyzes backup/archive status.

SQL Users

For SQL users, DBA/Query is a simple tool to learn and use. DBA/Query’s core engine conforms to relational database standards and technology. Database security provisions and access authorities are integrated into this tool, making usage very secure.

ISQL Users

DBA/Query overcomes the runtime limitations that ISQL users face. There is no need for left to right, up and down screen scrolling, associated with ISQL's constraining page width of 254 bytes and 19,995 selected rows of data. DBA/Query provides a streamlined interface data presentation, making data easily manageable and readable.

Usability

DBA/Query allows manipulation of ALLBASE/SQL data to be performed with ease, so memorization of complex SQL command syntax or column attributes for inserting or updating single rows of data is unnecessary. It is the perfect tool for troubleshooting databases and quickly fixing data. The report writer supports both raw and user defined formats. Not only is it comparable to any ALLBASE/SQL 4GL report writer on the market, but it also surpasses many of them. With DBA/Query, ALLBASE/SQL database environments and data can be controlled readily. Even database reporting capabilities are enhanced.

2. Installation and Update

Installation

1. Logon as a system manager:
   :HELLO DBAINSTL,MANAGER.SYS

2. Identify tape device:
   :FILE T;DEV=TAPE

3. Mount DBA/Query tape. Do not forget to turn the drive online.

4. Restore DBA/Query install job file into PUB.SYS:
   :RESTORE *T;DBAINSTL.PUB.SYS;SHOW

5. Insert system manager password to the install job file DBAINSTL.PUB.SYS, then stream it:
   :STREAM DBAINSTL.PUB.SYS

6. The DBA/Query install job first builds a new account LEETECH, then prompts for an install tape. If the tape is already mounted, simply turn the drive online. Otherwise, remount the tape.

7. The DBA/Query install job restores the files into PUB.LEETECH.

8. Purge the DBAINSTL.PUB.SYS upon successful completion of DBA/Query installation:
   :PURGE DBAINSTL.PUB.SYS

Update

1. Logon as a system manager:
   :HELLO DBAINSTL,MANAGER.SYS

2. Identify tape device:
   :FILE T;DEV=TAPE

3. Mount DBA/Query tape. Don't forget to turn the drive online.

4. Restore updated DBA/Query files into PUB.LEETECH:
   :RESTORE *T;@.@.LEETECH;SHOW

LeeTech Account

1. LEETECH account access security:
   (R,X:ANY;W,A,L:AC)
2. LEETECH account capability:
   AM,AL,GL,ND,SF,BA,IA,MR,DS,PH,PM
3. PUB.LEETECH group access security:
   (R,X:ANY;W,A,L,S:AC)
4. PUB.LEETECH group capability:
   BA,IA,MR,DS,PH,PM

3. Running

Running it Straight

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001      California Compensation
(1)>>
```

Using QRYINIT

DBA/Query can be set up to automatically execute a set of predefined commands. This Initial Command File is executed upon program initiating time. The default Initial Command File name is QRYINIT. A file equation can be used to redirect it.

Example of QRYINIT

```
:editor
/a
      1  # Open Database POLICY In MULTI mode
      2  b=policy.testdb.leetech
      3  # Access Table DBA.CLAIM
      4  s=dba.claim
      5  //
/k MYINIT,unn

:FILE QRYINIT=MYINIT
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001      California Compensation
QRYINIT(1)>> # Open Database POLICY In
QRYINIT(2)>> # MULTI mode
QRYINIT(3)>> b=policy.testdb.leetech
QRYINIT(4)>> # Access Table DBA.CLAIM
QRYINIT(5)>> s=dba.claim
(1)>>
```

Using STDIN=

Run DBA/Query with a STDIN= (standard input device) option. A DBA/Query script file can be used for standard session/job input files. The script file must be in ASCII and unnumbered format. It may contain only valid DBA/Query commands. The script file may also be used with XEQ commands.

Example of STDIN=

```
:editor
/a
      1  # Open Database POLICY
      2  b=policy.testdb.leetech
      3  # Access Table DBA.CLAIM
      4  s=dba.claim
      5  # Find All Claims
      6  f all
      7  # Report All Claims In Raw Format To
      8  # Terminal
      9  r all
      10 end
      11 //
/k MYCMD,unn

:RUN DBAQ.PUB.LEETECH;STDIN=MYCMD

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> # Open Database POLICY
(2)>> b=policy.testdb.leetech
(3)>> # Access Table DBA.CLAIM
(4)>> s=dba.claim
(5)>> # Find All Claims
(6)>> f all
4 entries qualified
(7)>> # Report All Claims In Raw Format To Terminal
(8)>> r all
Row #1
CLAIM_NO            = 'CLM0000004'
CLAIMANT_NO         = 'CMT0004001'
POLICY_NO           = 'POL1991004'
CLIENT_NO           = 'CLT0000004'
```

```
FILE_DATE           = 19910806
TYPE_OF_INJURY      = 'SB'
TOTAL_AMOUNT        = 900
PAID_AMOUNT         = 0

Row #2
CLAIM_NO            = 'CLM0000003'
CLAIMANT_NO         = 'CMT0005001'
POLICY_NO           = 'POL1991005'
CLIENT_NO           = 'CLT0000005'
FILE_DATE           = 19920804
TYPE_OF_INJURY      = 'FA'
TOTAL_AMOUNT        = 1000
PAID_AMOUNT         = 0
...
...
(9)>> end

END OF PROGRAM
```

4. Command Categories

The DBA/Query commands are grouped into 6 categories:

- Setup
- Data Manipulation
- Data Retrieval
- Report Writer
- Real-time Monitor
- Control

Setup

Setup commands are used to set values to DBA/Query control or environment variables, which include DATABASE NAME, TABLE/VIEW NAME, COLTYPE OPTION, ERRDIAG OPTION, HYPHEN OPTION, DOUBLEQUOTE OPTION and OUTPUT DEVICE.

Table 4-1. Setup Commands

B= **DATA-BASE=**	Opens a database.
S= **DATA-SETS**	Sets active table/view.
SET COLTYPE	Sets COLUMN type display option for row addition.
SET ERRDIAG	Sets command syntax diagnosis facility.
SET HYPHEN	Sets hyphen option to underscore.
SET DOUBLEQUOTE	Sets double quote option to single quote.
OUT=	Sets default output device.
DEFINE	Shows database environment and allows updating.
SHOW	Shows setup variables’ values.

Data Manipulation

Data manipulation commands are used to insert, update or delete table rows.

Table 4-2. Data Manipulation Commands

AD **ADD** **U ADD** **UPDATE ADD**	Inserts rows into the active table/view (by default) or the specified table/view.
DEL **DELETE** **U DELETE** **UPDATE DELETE**	Deletes rows retrieved by the last FIND command.
REAL **REPLACE** **U REPLACE** **UPDATE REPLACE**	Updates columns' values for rows retrieved by the last FIND command.

Data Retrieval

Data retrieval commands are used to retrieve data from a table/view.

Table 4-3. Data Retrieval Commands

F ALL **FIND ALL**	Finds all rows from the active table/view.
F **FIND**	Finds rows via selection criteria from the active table/view.
MULTIFIND **MU**	Finds rows via selection criteria from the compound table/view defined by JOIN.
MULTIFIND ALL **MU ALL**	Finds all rows from the compound table/view defined by JOIN.
JOIN	Defines the compound table/view used for multi-table/view data retrieval.
KEEP	Stores retrieved data into a pair of MPE files in DBA/Query raw format. Only DBA/Query report writer can use it.

Report Writer

LIST and REPORT ALL commands list the retrieved data in DBA/Query default report format, whereas the REPORT command invokes DBA/Query report writer to allow definition of a report format according to personal preferences.

Table 4-4. Report Writer Commands

LIST **R ALL** **REPORT ALL**	Lists retrieved rows in DBA/Query default report format.
R **REPORT**	Creates an interface to DBA/Query ad hoc report writer.
RW commands:	DBA/ Query report writer commands:
Header	Reports title and page headings.
Detail	Data item or register values.
Sort	Data item sort.
Group	Sorts item breaks.
Total	Total for group or report.
Edit	Edit masks and formats.
Register	Working registers' operation.
Output Control	Report output parameters.
File	Opens a data file for reports.

Real-time Monitor

Real-time monitor commands are used to inquire database real-time statistics and performance profiles.

Table 4-5. Real-time Monitor Commands

FORM COLUMN	Displays COLUMN attributes of a table/view.
FORM LOCK	Analyzes database locking and latching.
FORM RUNTIME	Displays database information regarding: • Database configuration. • Log file usage. • Database global usage analysis.
FORM SET	DBEFileSet utilization analysis. *No UPDATE STATISTICS*
FORM TABLE	Table/view row count.
FORM USER	Displays database information regarding: • MPE/XL session, program and database stored section inter-relationship analysis. • User process database access analysis.

Control

Control commands are DBA/Query level commands.

Table 4-6. Control Commands

COMMENT (#)	Ignored by DBA/Query.
EXIT (//)	Ends DBA/Query.
HELP **HELP command**	Invokes DBA/Query’s online help facility.
LISTREDO **REDO** **DO**	Conforms MPE/XL CI LISTREDO, REDO and DO command syntax.
X **XEQ**	Executes a DBA/Query script file (must be ASCII and unnumbered).

Command Usage

The database and the active table must be specified before performing data retrieval, data update, report generation or runtime status inquiry.

The database is also called DBEnvironment in ALLBASE/SQL. A DBEnvironment name is the name of the DBECON (DBEnvironment configuration) file.

There is no database level password in ALLBASE/SQL. The securities are set at data access, user and structure levels. Therefore, in ALLBASE/SQL, a successful database connection does not mean that data is necessarily accessible. It depends upon security provisions and access authorities.

5. DBA/Query Commands

ADD

```
AD            [[owner.]table|view]
ADD
U ADD
UPDATE ADD
```

Parameters

owner. The owner name of a table/view. The default owner is the MPE/XL logon (USER@ACCOUNT).

table or view The table/view name to which rows are added.

Discussion

TABLE or VIEW Inserts a row(s) into a table/view. DBA/Query uses the default active table/view if no parameters are specified.

PROMPT Each column name is prompted if COLTYPE is set to OFF. Column name and type are both prompted if COLTYPE is set to ON.

EXIT To end the ADD mode, either enter // or CONTROL-Y at the column prompt. The row exited from is not added to the table/view.

CONTINUATION A continuation '&' can be used for a CHARACTER type column. If it is used, a continuation prompt '>>' is displayed. Otherwise, DBA/Query is intelligent enough to capture all wrapped around lines and treats them as one input line. The CHARACTER types are CHAR, VARCHAR, DATE, TIME, DATETIME and INTERVAL.

INPUT IN QUOTE It is optional to use '' to surround the input for character type column.

Note: The input must be surrounded in ‘’ if an *&* (continuation) is part of the input data.

NUMERIC COLUMN

NUMERIC input, such as SMALLINT, INTEGER, DECIMAL and FLOAT must not be surrounded in ‘’. The input format conforms to ALLBASE/SQL standards. Signs (+/-) and period (.) are allowed.

NULL

Input is required for a NOT NULL column. NULL puts a null value into a column.

Example

Add one claim for policy POL1992001 to table DBA.CLAIM.

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> B=POLICY
(2)>> S=DBA.CLAIM
(3)>> ADD

CLAIM_NO            =>> CLM0000088
CLAIMANT_NO         =>> CMT0088001
POLICY_NO           =>> POL1992001
CLIENT_NO           =>> CLT0000008
FILE_DATE           =>> 19920216
TYPE_OF_INJURY      =>> CI
TOTAL_AMOUNT        =>> 1000
PAID_AMOUNT         =>> 800
CLAIM_NO            =>> //
1 entry added
(4)>>
```

DATA-BASE=

B= **DATA-BASE=**	**{*DBEnvironment*} [,SINGLE|,MULTI]**

Parameters

DBEnvironment The required database name.

SINGLE Opens the database in exclusive mode. It fails if any other users are concurrently accessing the same database.

MULTI Opens the database in share mode. It is not allowed if the database is configured to SINGLE mode. This is the default mode.

Discussion

Multiple database access is not supported. DATA-BASE= closes the old database and opens a new one.

Example

Open a POLICY database in exclusive mode.

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001      California Compensation
(1)>> B=POLICY,SINGLE
(2)>>
```

DATA-SETS=

S= **{[*owner*.] table|view}**
DATA-SETS=

Parameters

owner. The owner name of a table/view (optional). The default owner is the MPE/XL logon (USER@ACCOUNT).

table or view The table/view name.

Discussion

DATA-SETS= sets the default table/view, and it is used for all subsequent DBA/Query commands until another DATA-SETS= is specified. Only one active table/view can be designated the default at a time.

Example

Set a default table to DBA.CLAIM.

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001      California Compensation
...
(2)>> S=DBA.CLAIM
(3)>>
```

DEFINE

DEF[INE]

Discussion

DEFINE lists the state of the database environment and prompts for changes. After the prompt, a new parameter can be entered or press RETURN to keep the current value.

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> DEF
DATABASE            = ( )
DATABASE            =>> POLICY
MODE                =>> MULTI
TABLE/VIEW          =
TABLE/VIEW          =>> DBA.CLAIM
OUT                 = TERM
OUT                 =>> LP
(2)>> DEF
DATABASE            = POLICY
(POLICY.TESTDB.LEETECH)
DATABASE            =>>
TABLE/VIEW          = DBA.CLAIM
TABLE/VIEW          =>>
OUT                 = LP
OUT                 =>>
(3)
```

DELETE

DEL
DELETE
U DELETE
UPDATE DELETE

Discussion

DELETE deletes data entries by locating the last retrieval command from the default active table/view.

WARNING

For the DBA/Query (A.00.01) version, the DELETE command ignores the [#LIMIT=i] option of the FIND command. Thus, the deletion process can delete more rows than retrieved. This is usually caused by the inconsistency of ALLBASE/SQL versus TurboImage TRANSACTION concepts and implementation. Therefore, the DELETE confirmation prompt must be verified carefully.

CONFIRMATION

The confirmation displays the number of rows to be deleted. It must be verified before answering 'Y.' Any other responses are considered to be 'N' and the deletion process is canceled.

Example

Delete claim CLM0000088 off table DBA.CLAIM.

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> B=POLICY
(2)>> S=DBA.CLAIM
(3)>> F CLAIM_NO='CLM0000088'
1 entry qualified
```

```
(4)>> DELETE
Are you sure, 1 of entry will be deleted (Y/N)? Y
1 entry deleted
(5)>>
```

FIND

F **[#LIMIT=i,] {*selection criteria*}**
FIND

Parameters

#LIMIT= A reserved word.

i The maximum number of rows to be retrieved. Even if the total number of qualified rows in the active table/view exceeds the maximum limit, DBA/Query only retrieves the number of rows up to the maximum limit. Otherwise, it retrieves all the qualified rows within the maximum limit.

selection criteria The data retrieval criteria or the query definition. It conforms to ALLBASE/SQL SELECT command's WHERE clause syntax.

WHERE {Search Conditions}
ORDER BY {Column[ASC|DESC]} [,....]

Subquery is permitted in {Search Conditions}. The nested subquery supports up to 16 levels.

Discussion

Most ALLBASE/SQL supported Search Conditions are also supported by DBA/Query. The following predicates are those Search Conditions supported by DBA/Query:

Expression1 [NOT] BETWEEN Expression2 AND Expression3

Expression {=,<>,>,>=,<,<=} [Expression|Subquery]

EXISTS Subquery

Expression [NOT] IN {Subquery|(ValueList)}

Expression [NOT] LIKE {'Pattern'} ESCAPE {'EscapeChar'}

Column IS [NOT] NULL

Expression {=,<>,>,>=,<,<=} {ALL,ANY,SOME}
{Subquery|(ValueList)}

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>>  # Open Database POLICY
(2)>>  B=POLICY.TESTDB.LEETECH
(3)>>  # Access Table DBA.CLAIM
(4)>>  S=DBA.CLAIM
(5)>>  ######################################
(6)>>  # Find All Claims For Policy 'POL1992002'
(7)>>  ######################################
(8)>>  F POLICY_NO = 'POL1991002'
2 entries qualified
(9)>>  ######################################
(10)>> # Pattern Match                      #
(11)>> ######################################
(12)>> F POLICY_NO = 'POL%'
4 entries qualified
(13)>> ######################################
(14)>> # Order By                           #
(15)>> ######################################
(16)>> F POLICY_NO = 'POL%' ORDER BY CLAIM_NO

<CONTROL-Y>
3 entries have qualified, do you want to continue
searching (Y/N)? N
3 entries replaced
(17)>>
```

FIND ALL

```
F                        [#LIMIT=i]
F ALL
FIND ALL
```

Parameters

#LIMIT=

A reserved word.

i

The maximum number of rows to be retrieved. Even if the total number of qualified rows in the active table/view exceeds the maximum limit, DBA/Query only retrieves the number of rows up to the maximum limit. Otherwise, it retrieves all the qualified rows within the maximum limit.

Discussion

NO PARAMETER

DBA/Query retrieves all rows from the active table/view.

CONTROL-Y

Control-Y can be typed to suspend the retrieval process. During the break, DBA/Query displays how many rows have been retrieved and prompts for continuation. Enter ‘N’ to stop and retain those retrieved data, or enter any other key to resume the process.

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> B=POLICY
(2)>> S=DBA.CLAIM
(3)>> F
```

```
4 entries qualified
(4)>> F #LIMIT=2
2 entries qualified
(5)>>
```

FORM COLUMN

FO[RM] [COLUMN] **[[*owner.*] table|view]**

Parameters

owner. The owner name of a table/view.

table or view The table/view name.

Discussion

FORM COLUMN reports columns and the attributes of a table/view. If no parameter is specified, DBA/Query uses the active table/view as the default.

WILDCARD '@' Wildcard is NOT allowed.

CONTROL-Y N/A

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> B=POLICY
(2)>> S=DBA.CLAIM
(3)>> FO

FORM COLUMN       MON, FEB 17, 1992,  7:11 AM

TABLE/VIEW NAME: DBA.CLAIM

COLUMN NAME    COLUMN ATTRIBUTE  NULL/ NOT NULL
-----------    ----------------  --------------
CLAIM_NO       CHAR(10)          NOT NULL
CLAIMANT_NO    CHAR(10)          NOT NULL
POLICY_NO      CHAR(10)          NOT NULL
CLIENT_NO      CHAR(10)          NOT NULL
FILE_DATE      INTEGER           NOT NULL
```

```
TYPE_OF_INJURY CHAR(2)          NOT NULL
TOTAL_AMOUNT   INTEGER          NOT NULL
PAID_AMOUNT    INTEGER          NULL

(4)>> FO COLUMN DBA.ACTIVE_POLICY_CLIENT

FORM COLUMN      MON, FEB 17, 1992,  7:24 AM

TABLE/VIEW NAME: DBA.ACTIVE_POLICY_CLIENT

COLUMN NAME       COLUMN ATTRIBUTE   NULL/ NOT NULL
-----------------  ----------------   --------------
POLICY_NO         CHAR(10)           NOT NULL
CLIENT_NO         CHAR(10)           NOT NULL
CLIENT_NAME       CHAR(40)           NOT NULL
CLIENT_ADDRESS    CHAR(120)          NULL
POLICY_BEGIN_DATE INTEGER            NOT NULL
POLICY_END_DATE   INTEGER            NOT NULL
POLICY_TERM       CHAR(80)           NULL

(5)>>
```

FORM LOCK

FO[RM] LOCK

Discussion

FORM LOCK reports the program running status of users who are currently accessing the database. This report can be used to analyze database locking and resource management problems, including problems dealing with users holding up database resources, knowing which program is using most of the database services or if users are currently building an index in real-time.

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> B=POLICY.TESTDB.LEETECH
(2)>> S=DBA.CLAIM
(3)>> FO LOCK

FORM LOCK         MON, FEB 17, 1992, 7:24 AM

JOBNUM  PIN  PROGRAM           SQL INT FUNC  STATUS
------  ---  ----------------  ------------  -------
#S437   58   DBAQ.PUB.LEETECH  status        RUNNING
#S438   71   ISQL.PUB.SYS      drop index    RUNNING

(4)>>
```

Report Format

JOBNUM The MPE/XL session or job number.

PIN The MPE/XL process identification number in decimal form.

PROGRAM The name of the application program in the form of PROGRAM.GROUP.ACCOUNT

SQL INTERNAL FUNCTION The database current internal (DBCore) call function. Refer to HP's ALLBASE/SQL Database Administration Guide 8-6 to get a complete internal call function list.

STATUS RUNNING, LOCK WAIT, LATCH WAIT, PAGE WAIT, SERVICE WAIT or OTHER WAIT.

FORM RUNTIME

FO[RM] RUNTIME

Discussion

MAINTENANCE WORD FORM RUNTIME requests response to the database maintenance word. As a creator of the database, simply hit RETURN. Otherwise, a valid maintenance word must be provided.

This report contains DATABASE STATIC CONFIGURATION, LOG FILE ANALYSIS and DATABASE GLOBAL USAGE OVERVIEW.

CONFIGURATION Database controls block size, data buffer size, log buffer size and maximum number of transactions.

LOG FILE ANALYSIS This report is vital when using ALLBASE/SQL online database backup facility. It is created mainly for operating the monitor and controling the usage of log files to prevent any LOG FULL situation, which causes database shutdown. However, it is also suitable for a Database Administrator.

DYNAMIC DBA/Query shows log file information in DYNAMIC mode, which not only examines log files but also gets current status information from the connected database.

GLOBAL OVERVIEW This provides summary information about the global usage of a database. These statistics can be used to adjust the database setup or the application programs to improve the performance behavior of a database.

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> B=POLICY.TESTDB.LEETECH
(2)>> S=DBA.CLAIM
(3)>> FO RUNTIME
Maintenance Word:

FORM RUNTIME      MON, FEB 17, 1992,  7:24 AM

DATABASE CONFIGURATION:

Runtime Control Block Pages:   37
Data Buffer Pages:             50
Log Buffer Pages:              50
Maximum Transactions:          50

LOG FILE INFORMATION:

Archive Mode: OFF
Free Log Blocks: 211

LOG ID  LOGFILE/DUAL LOG   SIZE  USEABLE    BACKUP  ID BKUP STATUS
------- -----------------  ----- --------   ------- ---------------
*1      POLLOG01.TESTDB    250     YES        2    Not Required
 2      POLLOG02.TESTDB    150     YES        3    Not Required

DATABASE GLOBAL USAGE ANALYSIS:

Transaction begun:   17
Transaction ended:   16
Transaction abort:   3
Checkpoints:         1
Lock requests:       635
Shared latch reqs:   7060
Excl latch reqs:     13215
Page accesses:       5122
Page reads:          112
Page writes:         6
Log block reads:     493
Log block writes:    511
Log record writes:   2227
```

```
Number of Resets:    1

(5)>>
```

Report Format

ARCHIVE MODE — Can be ON or OFF, and is set only by the system. ON means that ARCHIVE LOGGING is in use.

FREE LOG BLOCKS — Indicates the total amount of log file space available in 512-byte blocks.

***** — Indicates the log file being accessed by the database.

LOG ID — Identifies a specific log file, and it is reusable when log files are purged and others are added. For dual logging, one LOG ID is assigned for both files.

LOG FILE — The name of a log file (MPE/XL file name).

DUAL LOG — The name of a dual log file (MPE/XL file name), if dual logging is used.

SIZE — The size of a log file in 512-byte records.

USEABLE — The usability of a log file.

BACKUP ID — Identifies a specific log file or a pair of files (dual logging) in an archive log sequence. It is unique and can never be reused.

BACKUP STATUS — NOT READY FOR BACKUP
This is the current file, which is not full yet.

READY FOR BACKUP
The file is full, and the database is now using a different file. Therefore, this file is ready to be stored with the STORELOG command in SQLUTIL.

BACKUP IS DONE
The file has already been backed up. It is ready to be reused.

BACKUP IS NOT REQUIRED
The database is in nonarchive mode, so the file does not need to be backed up.

FORM SET

FO[RM] SET[S] [*DBEFileSet*] [,[*lower*] ,[*upper*] ,[SCAN]]

Parameters

DBEFileSet

The name of DBEFileSet. The wildcard '@' is allowed.

lower

The lower bound %. It is used to exclude DBEFileSets whose utilization rate is smaller than the lower bound. The default is 0%. The DBEFileSet utilization rate used for lower bound comparison is the smallest one among three DBEFileSet types. They are TABLE, INDEX, and MIXED.

upper

The upper bound %. It is used to exclude DBEFileSets whose utilization rate is greater than thee upper bound. The default is 100%. The DBEFileSet utilization rate used for upper bound comparison is the largest one among three DBEFileSet types. They are TABLE, INDEX, and MIXED.

SCAN

A reserved word. If this option is specified, DBA/Query scans DBEFiles to analyze DBEFileSet's Capacity and Utilization real-time. Otherwise, DBA/Query gets DBEFileSet usage information directly from a system catalog, which contains values calculated by the most current UPDATE STATISTICS command.

Discussion

SCAN SPEED

The real time SCAN process runs very fast. It will not degrade database performance even when the system is heavily loaded.

WILDCARD ‘@’

The wildcard ‘@’ can be specified anywhere in the DBEFileSet name. DBA/Query will report space utilization for all matched DBEFileSets.

CONTROL-Y

The SCAN process breaks when Control-Y is hit. A prompt ‘Enter ‘N’ to stop the process. Continue?’ will be displayed. Hit ‘N’ to stop the process completely or press any key to continue scanning.

SCAN or NOT SCAN

The SCAN option triggers DBA/Query to perform DBEFiles’ scanning for DBEFileSet utilization analysis. It is REAL-TIME, therefore, no UPDATE STATISTICS is required. With the NO SCAN option, UPDATE STATISTICS is required to be able to report more accurate information.

The SCAN function can also be used to monitor a database’s temporary space allocation and system DBEFileSet utilization ratio. Both are very useful information for tuning database performance.

DBEFILE TYPES

MIXED, TABLE and INDEX.

MIXED

Stores both data and keys.

TABLE

Stores only data.

INDEX

Stores only keys.

In most instances, either use MIXED DBEFiles alone or use TABLE and INDEX DBEFiles within a DBEFileSet. MIXED DBEFiles mix key and data information into a DBEFileSet pool, which will eventually cause performance problems. Therefore, it is not recommended. TABLE and INDEX DBEFiles are much cleaner. They may be distributed or segregated onto different disk devices for performance tuning. The different DBEFiles cause a DBEFileSet to be full at different levels. For a MIXED only DBEFileSet, when all DBEFiles are full, the DBEFileSet is full. On the other hand, a TABLE and INDEX DBEFileSet is considered full, if either type of the DBEFiles is full.

HASH DBEFILE
The hash DBEFiles are always considered to be fully utilized.

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> B=POLICY.TESTDB.LEETECH
(2)>> S=DBA.CLAIM
(3)>> FO SETS

FORM SETS          MON, FEB 17, 1992,  7:24 AM

DBEFILESET NAME  TYPE   TOTAL PAGES  USED PAGES    FULL
---------------  -----  -----------  ----------  -------
CLAIM_SET        TABLE       50           50     100.00%
                 INDEX       50            0       0.00%
NEW_POLICY_SET   TABLE      150            2       1.33%
                 INDEX       50            3       6.00%
...
SYSTEM           MIXED      100           60      60.00%

(4)>> FO SET @POLICY@

FORM SETS          MON, FEB 17, 1992,  7:24 AM

DBEFILESET NAME  TYPE   TOTAL PAGES  USED PAGES    FULL
---------------  -----  -----------  ----------  -------
NEW_POLICY_SET   TABLE      150            2       1.33%
                 INDEX       50            3       6.00%
OLD_POLICY_SET   TABLE       50            0       0.00%
                 INDEX       50            3       6.00%
                 MIXED       50            2       4.00%

5)>> FO SET @POLICY@,,,SCAN
```

```
FORM SETS         MON, FEB 17, 1992,  7:24 AM

DBEFILESET NAME  TYPE   TOTAL PAGES  USED PAGES    FULL
---------------  -----  -----------  ----------   ------
NEW_POLICY_SET   TABLE      150           20       13.33%
                 INDEX       50           25       50.00%
OLD_POLICY_SET   TABLE       50           10       20.00%
                 INDEX       50           35       70.00%
                 MIXED       50           50      100.00%

(6)>>
```

FORM TABLE

FO[RM] TABLE   [[*owner.*]table|view] [,COUNT]

Parameters

owner.
The owner name of a table/view. Wildcard '@' is allowed to be used for name search.

table or view
The name of a table/view. Wildcard '@' is allowed to be used for name search.

COUNT
A reserved word. If this option is specified, DBA/Query counts the number of rows for real time tables and views. Otherwise, DBA/Query gets table row counts directly from the system catalog, which contains values calculated by the most current UPDATE STATISTICS command.

In the system catalog, view contains 0 row count.

The real time row count process may be lengthy, therefore, DO NOT use it when system load is heavy.

Discussion

DEFAULT
If no parameter is specified, the default table/view is the ACTIVE TABLE/VIEW, set by the 'DATA-SET=' command.

COUNT SPEED
The real time COUNT processing time depends heavily upon system load and the size of tables/views. For a large table, it may take a long while. Therefore, use it only when it is absolutely needed or when the database is not being heavily accessed.

WILDCARD '@'
The Wildcard '@' is allowed in the owner and table/view names.

CONTROL-Y The real time COUNT process breaks when Control-Y is hit. A prompt ‘Enter ‘N’ to stop the process. Continue?’ will be displayed. Hit ‘N’ to stop the process completely, or press any key to continue the counting.

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> B=POLICY.TESTDB.LEETECH
(2)>> S=DBA.CLAIM
(3)>> FO TABLE

FORM TABLE        MON, FEB 17, 1992,  7:24 AM

OWNER NAME TABLE/VIEW NAME       TOTAL ROWS  REMARK
---------- --------------------  ----------  ------------
DBA        CLAIM                     125     PUBLIC TABLE

(4)>> FO TABLE DBA.@

FORM TABLE        MON, FEB 17, 1992,  7:24 AM

OWNER NAME TABLE/VIEW NAME       TOTAL ROWS  REMARK
---------- --------------------  ----------  ------------
DBA        ACTIVE_POLICY              25     PUBLIC TABLE
DBA        ACTIVE_POLICY_CLIENT       38     VIEW
DBA        CLAIM                     125     PUBLIC TABLE
DBA        CLAIMANT                   77     PUBLIC TABLE
DBA        CLAIMANT_PAYMENT           23     VIEW
...

(5)>> FO TABLE DBA.@,COUNT

FORM TABLE        MON, FEB 17, 1992,  7:24 AM

OWNER NAME TABLE/VIEW NAME       TOTAL ROWS  REMARK
---------- --------------------  ----------  ------------
DBA        ACTIVE_POLICY              32     PUBLIC TABLE
DBA        ACTIVE_POLICY_CLIENT       27     VIEW
DBA        CLAIM                     133     PUBLIC TABLE
DBA        CLAIMANT                   88     PUBLIC TABLE
DBA        CLAIMANT_PAYMENT          13     VIEW
...
```

FORM USER

```
FO[RM] USER     [session,] [user.account]

FO[RM] USER     #Snnnn | #Jnnnn
```

Parameters

session — The MPE/XL logon session name. Wildcard ‘@’ is allowed. The default is ALL sessions which contain programs connecting to the specified database.

user.account — The MPE/XL logon user and account name. Wildcard ‘@’ is allowed. The default is ALL users of ALL accounts which contain programs connecting to the specified database.

#S*nnnn* **|** #J*nnnn* — A session number (#Snnnn) or a batch job number (#Jnnnn). This is a required parameter. Wildcard ‘@’ is NOT allowed.

Discussion

The FORM USER command has two forms. The first form presents summaries about a MPE user, the application program and the database access state. The other one is used for a detailed analysis of a particular MPE user.

MPE USER — The MPE user contains a process(es) which accesses the database specified by the DATA-BASE= command.

APPLICATION PROGRAM — The application programs are connected to the database specified by the DATA-BASE= command.

PROGRAM STATE — They include EXEC, READY, NULL and UNKNOWN. These states are displayed when the application program is executing nondatabase related functions or when waiting for I/O.

ALLBASE/SQL STATE They are LOCK*, LATCH1*, LATCH2*, LATCH3*, LATCH4* and BUFFER*. These states are displayed when the application program is performing database related functions or is waiting for database related resources.

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> B=POLICY.TESTDB.LEETECH
(2)>> S=DBA.CLAIM
(3)>> FO USER

FORM USER         MON, FEB 17, 1992,  7:24 AM

JOB
NUM   JOB NAME                      PROGRAM                        PIN   QPRI    STATE
----- ----------------------------- ------------------------------ ----  ------  ---------
#S64  LEE,MGR.LEETECH               DBAQ.PUB.LEETECH               51    C153    EXEC
#S71  DANA,MANAGER.TMS              TMS0000P.TMSPRG.TMS            77    C141    LOCK*
#S71  DANA,MANAGER.TMS              TMS0007P.TMSPRG.TMS            78    C150    LOCK*
#S72  BOB,TDAUSER.TMS               TMS0005P.TMSPRG.TMS            81    C140    LOCK*
#S73  CAROL,FTCUSER.TMS             TMS0000P.TMSPRG.TMS            43    C150    LOCK*
#S74  GWEN,GPUSER.TMS               TMS0009P.TMSPRG.TMS            75    C143    WAIT
......
......

(4)>> FO USER @,@USER.TMS

FORM USER         MON, FEB 17, 1992,  7:24 AM




JOB
NUM   JOB NAME                      PROGRAM                        PIN   QPRI    STATE
----- ----------------------------- ------------------------------ ----  ------  ---------
```

```
#S72  BOB,TDAUSER.TMS      TMS0005P.TMSPRG.TMS  81   C140   LOCK*
#S73  CAROL,FTCUSER.TMS    TMS0000P.TMSPRG.TMS  43   C150   LOCK *
#S74  GWEN,GPUSER.TMS      TMS0009P.TMSPRG.TMS  75   C143   WAIT
......
......
```

```
(5)>> FO USER #S74

FORM USER         MON, FEB 17, 1992,  7:24 AM

JOB
NUM    STDIN    STDOUT      INTRODUCED                            JOB NAME
------ -------- ----------- ------------------------------------  -----------------------
#S74   113      113         MON, FEB 17, 1992, 7:24 AM            GEN,GPUSER.TMS

                ACCESS   LOG      TEMP     PERM
PIN   STATE     PAGES    BYTES    PAGES    PAGES    PROGRAM
----  --------- -------- -------- -------- -------- ------------------------------
75    WAIT         245        0       45       18     TMS0009P.TMSPRG.TMS

(6)>>
```

JOIN

J[OIN]	**column equivalence [,column equivalence] [;table or view equivalence [,table or view equivalence]...] [end]**

Parameters

column — The name of the column.

table or view — The table/view name.

column equivalence — It has the form:

```
column [@] to [@] column
```

table or view equivalence — It has the form:

```
table = table
view = view
```

Discussion

JOIN allows the retrieval and reporting of data item values from multiple tables/views. Once a compound data table/view has been defined with the JOIN command, a MULTIFIND or a MULTIFIND ALL command must be entered to actually access the desired items.

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> #Open Database POLICY
(2)>> B=POLICY.TESTDB.LEETECH
(3)>> JOIN DBA.CLAIM.CLIENT_NO TO
DBA.CLIENT.CLIENT._NO
(4)>> MULTIFIND
4 entries qualified
```

KEEP

KEEP ***{filename}***

Parameters

filename This is the name of the header file.

Discussion

This command saves the data entries located by the last retrieval command into a set of permanent files in DBA/Query raw format.

Note: KEEP can only be used by a DBA/Query Report Writer.

The <filename> is the name of the header file. Its corresponding data file is saved with a unique name generated by DBA/Query in the form of <FMAnnnnn>. Both files are placed into the current home group.

RW FILE COMMAND The FILE command in Report Writer can be used to access the files that were saved via the KEEP command.

Note: The FILE command must be the first and the only one for each report request.

Example

```
      :RUN DBAQ.PUB.LEETECH

      DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
      1992.
      License No. 000001      California Compensation
      (1)>> B=POLICY
      (2)>> S=DBA.CLAIM
      (3)>> F POLICY='POL0001'
      4 entries qualified
      (4)>> KEEP CLAIM001
Header=CLAIM001.DB.LEETECH, Data=FMA14463.DB.LEETECH
      (5)>> REPORT
      RW>> FILE CLAIM001
      RW>> H1,"DATE:",5
      RW>> H1,DATE,15
      RW>> H1,"CLAIMS FOR POLICY - POL0001",60
      RW>> H3,"CLIENT",9
      RW>> H3,"CLAIM NO",23
      RW>> H4,"=========",10
      RW>> H4,"=========",25
      RW>> D1,CLIENT_NO,10
      RW>> D1,CLAIM_NO,25
      RW>> END

      DATE:07/24/1991   CLAIMS FOR POLICY - POL0001

      CLIENT NO       CLAIM NO
      ----------------  ---------------
      CLT0000001      CLM0000001
      CLT0000008      CLM0000002
      ....
      ....
```

MULTIFIND

MULTIFIND [#LIMIT=i,] [*selection criteria*]

Parameters

#LIMIT= A reserved word.

i The maximum number of rows to be retrieved. Even if the total number of qualified rows in the active table/view exceeds this maximum limit, DBA/Query only retrieves the number of rows up to the maximum limit. Otherwise, it retrieves all the qualified rows.

selection criteria The data retrieval criteria or the query definition. It conforms to ALLBASE/SQL SELECT command's WHERE clause syntax.

Discussion

MULTIFIND retrieves data rows from compound data tables/views according to the selection critieria. This command only retrieves the rows that belong to the compound tables/views previously defined by command JOIN.

AMBIGUOUS COLUMN When a column that appears in a selection critieria belongs to different tables/views, specify the full path of the column to avoid ambiguous criteria.

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001      California Compensation
(1)>> B=POLICY
(2)>> JOIN DBA.CLAIM.CLIENT_NO TO
DBA.CLIENT.CLIENT_NO
(3)>> MULTIFIND
```

```
      4 entries qualified
      (4)>> MULTIFIND TOTAL_AMOUNT=1200
      1 entry qualified
```

MULTIFIND ALL

MU[LTIFIND] ALL   [#LIMIT=i]

Parameters

#LIMIT=   A reserved word.

i   The maximum number of rows to be retrieved. Even if the total number of qualified rows in the active table/view exceeds this maximum limit, DBA/Query only retrieves the number of rows up to the maximum limit. Otherwise, it retrieves all the qualified rows.

Discussion

MULTIFIND ALL retrieves all data rows from the compound tables/views specified by the most recent JOIN command.

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No.  000001      California Compensation
(1)>> B=POLICY
(2)>> JOIN DBA.CLIENT.CLIENT_NO TO
DBA.CLAIM.CLIENT_NO
(3)>> MULTIFIND ALL
4 entries qualified
(4)>> MULTIFIND ALL #LIMIT=2
2 entries qualified
```

OUT=

```
OUT=                    { LP | TERM }
```

Parameters

LP

Set the output device to the printer. The default spooler file name is QSLIST.

TERM

Set the output device to the terminal. The default name is TERM.

Discussion

MPE/XL FILE commands can be used to redirect the default output file QSLIST to a more meaningful spooler name or to a permanent file.

The command OUT= affects the output generated by the LIST and REPORT ALL commands only.

The OUT= setting is not carried over to REPORT WRITER. In REPORT WRITER, the command OUT= must be used to specify output destination again. Make sure the two settings are not in conflict with each other. Thus when returning back to DBA/Query's command program, the OUT= setting remains unchanged.

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> # Open Database POLICY
(2)>> B=POLICY.TESTDB.LEETECH
(3)>> # Access Table DBA.CLAIM
(4)>> S=DBA.CLAIM
(5)>> # Find All Claims
(6)>> F ALL
4 entries qualified
(7)>> # Report All Claims In Raw Format To Terminal
(8)>> R ALL
```

```
Row #1
CLAIM_NO            = 'CLM0000004'
CLAIMANT_NO         = 'CMT0004001'
POLICY_NO           = 'POL1991004'
CLIENT_NO           = 'CLT0000004'
FILE_DATE           = 19910806
TYPE_OF_INJURY      = 'SB'
TOTAL_AMOUNT        = 900
PAID_AMOUNT         = 0

Row #2
CLAIM_NO            = 'CLM0000003'
CLAIMANT_NO         = 'CMT0005001'
POLICY_NO           = 'POL1991005'
CLIENT_NO           = 'CLT0000005'
FILE_DATE           = 19910804
TYPE_OF_INJURY      = 'FA'
TOTAL_AMOUNT        = 1000
PAID_AMOUNT         = 0
...
...
(7)>> # Set Output Device to Printer
(8)>> OUT=LP
(9)>> # Report All Claims In Raw Format To QSLIST
(10)>> R ALL
(11)>> :LISTSPF

SPOOLID  JOBNUM  FILEDES  PRI  COPIES  DEF  STATE ...
-------  ------  -------  ---  ------  ---  ----- ---
#055     S428    QSLIST    8     1     LP   READY ...
...
...
(12)>> :PRINT 055.OUT.HPSPOOL
PRINT 055.OUT.HPSPOOL
PRINT OUTPUT HAS BEEN TRUNCATED TO 270 CHARACTERS.
...


Row #1
CLAIM_NO            = 'CLM0000004'
CLAIMANT_NO         = 'CMT0004001'
POLICY_NO           = 'POL1991004'
CLIENT_NO           = 'CLT0000004'
FILE_DATE           = 19910806
TYPE_OF_INJURY      = 'SB'
```

```
TOTAL_AMOUNT        = 900
PAID_AMOUNT         = 0

Row #2
CLAIM_NO            = 'CLM0000003'
CLAIMANT_NO         = 'CMT0005001'
POLICY_NO           = 'POL1991005'
CLIENT_NO           = 'CLT0000005'
FILE_DATE           = 19910804
TYPE_OF_INJURY      = 'FA'
TOTAL_AMOUNT        = 1000
PAID_AMOUNT         = 0
...
...
(13)>>
```

REPLACE

```
REPL                column=value [,column=value] ....
REPLACE
U REPLACE
UPDATE REPLACE
```

Parameters

column — The name of the column to be modified.

value — The column replacing value.

It must be surrounded in single quotes ‘_’ for CHAR, VARCHAR, DATE, TIME, DATATIME, and INTERNAL types of columns.

It must be in numeric form for SMALLINT, INTEGER, DECIMAL, and FLOAT types of columns.

NULL is for a null value. The column must allow a null value.

Discussion

WARNING — For DBA/Query version (A.00.01), the update function ignores the [#LIMIT=i] option of the FIND command. Thus the update process can update more rows than received. This is usually caused by the inconsistency of ALLBASE/SQL versus TurboImage TRANSACTION concepts and implementation. Therefore, verify the update confirmation prompt carefully.

SQL SYNTAX — The parameter syntax conforms to ALLBASE/SQL UPDATE command’s SET clause syntax. Several columns of the same table can be updated with this command.

COLUMN TYPE The input data type must be compatible with the column’s data type.

HASH KEY HASH KEY cannot be updated.

Example

Update CLAIMANT_NO from 'CMT0002001' to 'CMT0002666' for all claims of policy 'POL1991002.'

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> # Open Database POLICY
(2)>> B=POLICY.TESTDB.LEETECH
(3)>> # Access Table DBA.CLAIM
(4)>> S=DBA.CLAIM
(5)>> # Find All Claims For Policy 'POL1992002'
(6)>> F POLICY_NO = 'POL19911002'
2 entries qualified
(7)>> # Update CLAIMANT_NO
(8)>> REPL, CLAIMANT_NO = 'CMT0002666'
Are you sure, 2 of entries will be replaced (Y/N)?
Y
2 entries replaced
(9)>>
```

REPORT

R **REPORT**	**[*report script file*]**

Parameters

report script file — An ASCII and an unnumbered MPE/XL file that contains a set of report writer commands for generating a specific report. At run time, the DBA/Query report writer reads and executes commands directly from this file instead of getting input interactively from the terminal. The maximum size of a script file record is 80 characters. Any characters beyond the 80th are ignored by the DBA/Query script writer.

Discussion

REPORT connects the DBA/Query main process and ad hoc report writer. All of report writer commands are 100% compatible to TurboImage/QUERY.PUB.SYS syntax (refer to HP QUERY/V Reference Manual 3-105).

THE DBA/Query Report Writer (RW) prompt is RW>>.

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> B=POLICY.TESTDB.LEETECH
(2)>> S=DBA.CLAIM
(3)>> F POLICY='POL0001'
5 entries qualified
(4)>> R
RW>> H1,"DATE:",5
RW>> H1,DATE,15
RW>> H1,"CLAIMS FOR POLICY - POL0001",60
RW>> H3,"CLIENT NO",9
RW>> H3,"CLAIM NO",23
RW>> H4,"========",10
RW>> H4,"========",25
RW>> D1,CLIENT_NO,10
RW>> D1,CLAIM_NO,25
RW>> END

DATE:07/24/1991   CLAIMS FOR POLICY - POL0001

CLIENT NO         CLAIM NO
==========        ==========
CLT0000001        CLM0000001
CLT0000008        CLM0000002
....
:editor
/a
      1   H1,"DATE:",5
      2   H1,DATE,15
      3   H1,"CLAIMS FOR POLICY - POL0001",60
      4   H3,"CLIENT NO",9
      5   H3,"CLAIM NO",23
      6   H4,"==========",10
      7   H4,"==========",25
      8   D1,CLIENT_NO,10
      9   D1,CLAIM_NO,25
      10  END
      11  //

/k CLAIMRPT,unn

:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
...
(14)>> S=DBA.CLAIM
(15)>> F POLICY='POL0001'
5 entries qualified
(16)>> R CLAIMRPT
RW>> H1,DATE,15
RW>> H1,"CLAIMS FOR POLICY - POL0001",60
RW>> H3,"CLIENT NO",9
RW>> H3,"CLAIM NO",23
RW>> H4,"==========",10
RW>> H4,"==========",25
RW>> D1,CLIENT_NO,10
RW>> D1,CLAIM_NO,25
RW>> END

DATE:07/24/1991   CLAIMS FOR POLICY - POL0001

CLIENT NO         CLAIM NO
==========        ==========
CLT0000001        CLM0000001
CLT0000008        CLM0000002
....
```

REPORT ALL

LIST
R ALL
REPORT ALL

Discussion

REPORT ALL prints data values of entries located by the last retrieval command in DBA/Query default report format.

QSLIST

QSLIST is the default output file name, if command OUT=LP is specified. Otherwise, the terminal is the default output device. MPE/XL FILE command can be used to redirect QSLIST to another name or to a permanent file.

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> # Open Database POLICY
(2)>> B=POLICY.TESTDB.LEETECH
(3)>> # Access Table DBA.CLAIM
(4)>> S=DBA.CLAIM
(5)>> # Find All Claims
(6)>> F ALL
5 entries qualified
(7)>> # Report All Claims In Raw Format To Terminal
(8)>> R ALL
Row #1
CLAIM_NO            = 'CLM0000004'
CLAIMANT_NO         = 'CMT0004001'
POLICY_NO           = 'POL1991004'
CLIENT_NO           = 'CLT0000004'
FILE_DATE           = 19910806
TYPE_OF_INJURY      = 'SB'
TOTAL_AMOUNT        = 900
PAID_AMOUNT         = 0
```

```
Row #2
CLAIM_NO            = 'CLM0000003'
CLAIMANT_NO         = 'CMT0005001'
POLICY_NO           = 'POL1991005'
CLIENT_NO           = 'CLT0000005'
FILE_DATE           = 19910804
TYPE_OF_INJURY      = 'FA'
TOTAL_AMOUNT        = 1000
PAID_AMOUNT         = 0
...
...
(7)>> # Set Output Device To Printer
(8)>> OUT=LP
(9)>> # Report All Claims In Raw Format To QSLIST
(10)>> R ALL
(11)>> :LISTSPF

SPOOLID  JOBNUM FILEDES  PRI  COPIES  DEF  STATE ...
-------  ------ -------  ---  ------  ---  ----- ---
#O55     S428   QSLIST    8     1     LP   READY ...
...
...
(12)>> :PRINT O55.OUT.HPSPOOL
PRINT O55.OUT.HPSPOOL
PRINT OUTPUT HAS BEEN TRUNCATED TO 270 CHARACTERS.
...

Row #1
CLAIM_NO            = 'CLM0000004'
CLAIMANT_NO         = 'CMT0004001'
POLICY_NO           = 'POL1991004'
CLIENT_NO           = 'CLT0000004'
FILE_DATE           = 19910806
TYPE_OF_INJURY      = 'SB'
TOTAL_AMOUNT        = 900
PAID_AMOUNT         = 0

Row #2
CLAIM_NO            = 'CLM0000003'
CLAIMANT_NO         = 'CMT0005001'
POLICY_NO           = 'POL1991005'
CLIENT_NO           = 'CLT0000005'
```

```
FILE_DATE           = 19910804
TYPE_OF_INJURY      = 'FA'
TOTAL_AMOUNT        = 1000
PAID_AMOUNT         = 0
...
...
(13)>>
```

REPORT WRITER

Summary of Report Writer Commands

Table 5-1. Report Writer Commands

HEADER	Prints title, column headings, page numbers, time of day, and the date at the top of each report page.
DETAIL	Prints data item or registers values or a character string in the column position specified.
SORT	Sorts data based on the value of a specified data column.
GROUP	Prints a data item value or a character string whenever the value of an appropriate “sort item” changes.
TOTAL	Prints count, average, or totals for logical groups or an entire report.
EDIT	Describes edit masks used to punctuate Group, Detail, or Total fields.
REGISTER	Specifies an operation to be executed in Register n.
OUTPUT CONTROL	Specifies the report output parameters.
FILE	Opens a data file saved by DBA/Query’s KEEP command for reports.

Summary of Report Writer Command Parameters

Table 5-2. Report Writer Command Parameters

PRINT POSITION	Determines the rightmost print position (column number) for the PRINT ELEMENT. For character data, this is the last character of the last word. For numeric data, it is the position of the least significant digit.
SPACE A[number]	Space [number] lines after printing the report line. If [number] is omitted, one line is spaced.
SPACE B[number]	Space [number] lines before printing the report line. If [number] is omitted, one line is spaced.
number	The number of lines to be spaced (from 1 to 5).
SKIP {A|B}	Skips to the top of the next report page after printing the report line (SKIP A) or before printing the report line (SKIP B).
E{number|Z}	Indicates that an edit mask defined in the identically numbered edit statement (Enumber) is used to punctuate a value or, if the letter Z is used, that leading zeros are to be suppressed. In the latter case, no edit statement is required.

Skipping and Spacing

When paging is in effect, the following rules govern skipping and spacing:

1. If a SKIP B and a SPACE B are both associated with the same output line, the SKIP B is processed before the SPACE B.
2. If not enough lines remain on the current page to satisfy a SPACE B, then a page will be form fed before the spacing takes effect.
3. If a SKIP A and a SPACE A are both associated with the same output line, the SPACE A is ignored.
4. If not enough lines remain on the current page to satisfy a SPACE A, then it will be treated as a SKIP A.

5. A SKIP A on the last line of the report will not be executed.
6. SPACE A and SPACE B are allowed on the same REPORT statement. The SPACE B is done first, then the SPACE A.

REPORT - Detail

**D*detail number*, print element, print position [,SPACE A[*number*]]
[,SPACE B[*number*]] [,SKIP{A|B}] [,E{*number*|Z}]**

Parameters

detail number	An integer from 1 to 99. If the number is omitted, the <print element> is printed on a group line when any control break occurs. The lowest numbered statement is printed first and others follow in a numeric sequence. Detailed statements with the same number are printed on the same line. Therefore, information from a single data entry can be printed on up to 100 separate lines.
print element	(1) The name of a single column. The value of the specified data column is printed. (2) The content of a register (R*n*). If a register is specified, the data in the register is printed. Both the letter (R) and the number (*n*) must be entered. (3) A series of characters enclosed in quotation marks. The characters are printed without the surrounding quotation marks, and are printed once for each reported entry.
print position SPACE A[*number*] SPACE B[*number*] SKIP {A|B} E{*number*|Z}	Refer to Table 5-2. Summary of Report Writer Command Parameters.

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> B=POLICY.TESTDB.LEETECH
(2)>> S=DBA.CLAIM
(3)>> F POLICY='POL0001'
5 entries qualified
(4)>> R
RW>> H1,"DATE:",5
RW>> H1,DATE,15
RW>> H1,"CLAIMS FOR POLICY - POL0001",60
RW>> H3,"CLIENT NO",9
RW>> H3,"CLAIM NO",23
RW>> H4,"==========",10
RW>> H4,"==========",25
RW>> D1,CLIENT_NO,10
RW>> D1,CLAIM_NO,25
RW>>END

DATE:07/24/1991   CLAIMS FOR POLICY - POL0001

CLIENT NO         CLAIM NO
==========        ==========
CLT0000001        CLM0000001
CLT0000008        CLM0000002
....
```

REPORT - Edit

E*detail number*, "edit mask"

Parameters

detail number

An integer from 0 to 9 that identifies the edit statement. A report can have at most ten edit statements, each with a unique number.

"edit mask"

Consists of characters from 1 to 20 (if the mask is to edit numeric data item values), or from 1 to the maximum length of the data item, not to exceed the maximum record length of the output device (if the mask is to edit alphanumeric data item values). In either case, the characters must be surrounded by quotation marks. The length of the edit mask determines the length of the printed output field.

Summary of Edit Mask Characters

Table 5-3. Edit Mask Characters

Alphanumeric Edit Mask:	
X	Each X in the edit mask is replaced by an alphanumeric character from the data item value in left to right order.
Numeric Edit Mask:	
9	Each 9 in the edit mask is replaced by a decimal digit from the data item value in the corresponding position of the output field.
Z	Z is a zero suppression place holder. A Z in the edit mask is replaced with a decimal digit from the data item value in the corresponding position of the output field. If the data item value digit under consideration is a zero appearing to the left of the leftmost significant digit, then DBA/Query inserts a blank in the output field, and all other zeros to the left of the significant digit are replaced by a blank in the output field.
*****	* is an asterisk place holder. An * in the edit mask acts like a Z with the exception that leading zeros in the data item value are replaced with * in the output field.
$	$ is the dollar sign place holder. A $ in the edit mask acts just like a Z, except that the first zero in the data item value to the left of the leftmost significant digit is replaced with $ in the output field. All zeros to the left of the first leading zero are replaced with blanks in the output field.

Table 5-3. Edit Mask Characters (continued)

Numeric Edit Mask:	
CR	CR is a sign character, and it always appears in two rightmost positions of the edit mask. If the data item value is negative, DBA/Query prints the two characters (CR) in the first two rightmost positions of the output field. If the data item value is positive, DBA/Query prints two blank characters in place of the CR. No characters from the data item value are ever placed in the first two rightmost positions of the output field.
-	- is a sign character and it acts same as the CR characters. If the data item value is negative, then DBA/Query prints the minus sign (-) in the rightmost position of the output field. If the data item value is positive, then DBA/Query prints a blank in the place of the minus. No character from the data item value is ever placed in the rightmost position of the output field.
Insertion characters	Insertion characters consist of any ASCII printing characters not previously mentioned. Insertion characters are printed in the output field in the same position they appear in the edit mask. Any insertion character appearing in the edit mask to the left of the leftmost significant digit of the data item value is replaced with blanks or an asterisk, depending upon which zero suppression character is specified in the edit mask. Only one decimal point can appear in an edit mask.

Sample Output Using Edit Masks

DATA ITEM VALUE	EDIT MASK	PRINTED RESULT
LeeTech Software Inc.	"XXXXXXX"	LeeTech
0059	"$$$,999"	$059
001024	"ZZZ,ZZZ"	1,024
-0010555	"$$,$$$.99CR"	$105.55CR
00010555	"$$,$$$.99CR"	$105.55
-0010555	"$$,$$$.99-"	$105.55-
15039250	"$,$$$,$$$.99CR"	$150,392.50
00049	"*****"	***49
044240474	"999-99-9999"	044-24-0474
-2145	"$,$$$.99"	$21.45
0.00	"ZZZZZ9"	0
123456.78	"999999"	123456
.0034567	".999999"	.003456

Figure 5-1. Sample Edit Mask Output

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> B=POLICY.TESTDB.LEETECH
(2)>> S=DBA.CLAIM
(3)>> F POLICY='POL0001'
5 entries qualified
(4)>> R
RW>> H1,"DATE:",5
RW>> H1,DATE,15
RW>> H1,"CLAIMS FOR POLICY - POL0001",60
RW>> H3,"CLIENT NO",9
RW>> H3,"CLAIM NO",23
RW>> H3,"FILE DATE",40
RW>> H4,"==========",10
RW>> H4,"==========",25
RW>> H4,"==========",40
RW>> E1,"99/99/9999"
RW>> D1,CLIENT_NO,10
RW>> D1,CLAIM_NO,25
RW>> D1,FILE DATE,E1,25
RW>> END

DATE:07/24/1991   CLAIMS FOR POLICY - POL0001

CLIENT NO     CLAIM NO      FILE DATE
==========    ==========    ==========
CLT0000001    CLM0000001    01/12/1991
CLT0000008    CLM0000002    12/21/1990
....
```

REPORT - Group

G*level*, print element, print position [,SPACE A[*number*]] [,SPACE B[*number*]] [,SKIP{A|B}] [,E{*number*|Z}]

Parameters

level	An integer from 1 to 10 that corresponds to the level of a sort statement.
print element	(1) The name of a single column. The value of the specified data column is printed. (2) The content of a register (R*n*). (3) A series of characters enclosed in the quotation marks. The quotation marks must be specified, but DBA/Query will strip them when it prints the characters.
print position SPACE A[*number*] SPACE B[*number*] SKIP {A|B} E{*number*|Z}	Refer to Figure-2 Summary of Report Writer Command Parameters.

Discussion

Each control break occurs as the result of a sort statement labeled from 1 to 10. When the control break occurs, the group statement with the same number as the sort statement, prints the information specified. Whenever a control break occurs, all group statements with a number equal to or less than the level of the sort statement causing the break will print a value and/or a series of characters. All group statements print on the same line. Since a control break always occurs at the very beginning of the report, all group statements print their contents before any detail statements are executed.

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001   California Compensation
(1)>> B=POLICY.TESTDB.LEETECH
(2)>> S=DBA.CLAIM
(3)>> F POLICY='POL0001'
5 entries qualified
(4)>> R
RW>> H1,"DATE:",5
RW>> H1,DATE,15
RW>> H1,"CLAIMS FOR POLICY - POL0001",60
RW>> H3,"CLIENT NO",9
RW>> H3,"CLAIM NO",23
RW>> H4,"==========",10
RW>> H4,"==========",25
RW>> S1,CLIENT_NO
RW>> S1,CLAIM_NO
RW>> G1,CLIENT_NO,10
RW>> D1,CLAIM_NO,25
RW>> END

DATE:07/24/1991   CLAIMS FOR POLICY - POL0001

CLIENT NO         CLAIM NO
--------------------      --------------------
CLT0000001
                  CLM0000001
                  CLM0000005
CLT0000006
                  CLM0000004
CLT0000008
                  CLM0000002
                  CLM0000003
```

REPORT - Header

H*header number*, print element, print position [,SPACE A[*number*]] [,SPACE B[*number*]] [,E{*number*|Z}]

Parameters

header number

An integer from 1 to 9. Up to nine lines of header information can be printed in addition to blank lines created by spacing before and after non-blank lines. Header information with the same header number is printed on the same line. The lowest-numbered header statement is printed first, and the next-lowest numbered statement is printed next. Header statements do not have to be consecutively numbered.

print element

(1) The name of a single column. The value of the specified data column is printed.

(2) The character enclosed in quotation marks. The characters are stripped of the surrounding quotation marks and are printed.

(3) PAGENO, DATE, or TIME

PAGENO	Consecutively numbers each page of the report.
DATE	Prints the date in the form: MM/DD/YYYY.
TIME	Prints the time in the form: HH:MM:SS.

print position
SPACE A*number*
SPACE B*number*
E{*number*|Z}

Refer to Figure-2 Summary of Report Writer Command Parameters.

Discussion

A header can contain up to 9 lines of information and any number of blank lines as long as it does not exceed the page size as defined by the output control statement LINES=.

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001      California Compensation
(1)>> B=POLICY.TESTDB.LEETECH
(2)>> S=DBA.CLAIM
(3)>> F POLICY='POL0001'
5 entries qualified
(4)>> R
RW>> H1,"DATE:",5
RW>> H1,DATE,15
RW>> H1,"CLAIMS FOR POLICY - POL0001",60
RW>> H3,"CLIENT NO",9
RW>> H3,"CLAIM NO",23
RW>> H4,"==========",10
RW>> H4,"==========",25
...
```

```
DATE:07/24/1991   CLAIMS FOR POLICY - POL0001

CLIENT NO         CLAIM NO
--------------------         --------------------
...
```

REPORT - Output Control

Table 5-4. Output Control

LINES=integer	Specifies the number of lines per report page. <integer> can be anywhere between 10 and 32,767. If <integer> is 0, the page size is infinite.
NOPAGE	Suppresses page advancing at the beginning of each page (no margins are provided at the top and bottom of each page). All SKIP options are ignored, and all SPACE options are performed unconditionally.
[OUT=]LP	Sends the report output to the QSLIST device. It applies only to the current report.
PAUSE	Causes the report output to pause after each page is completed. Press RETURN to continue. PAUSE adds an extra line as RETURN is pressed. PAUSE is ignored in job mode or when output is send to QSLIST.
UNIFYDETAIL	Causes page eject when a particular block of detailed statements would split between two pages. A block is all the detailed statements for each entry or compound entry.

Default Output Control

1. 60 lines per page.
2. Page advancing at the beginning of each report page. On the terminal, the page advancing appears as 6 line feeds.
3. Output printed on the $STDLIST device (the terminal in session mode and line printer in job mode).
4. No pauses while the report is being printed.

REPORT - Register

R*number*, { L[OAD] | A[DD] | S[UBTRACT] | M[ULTIPLY] | D[IVIDE] }
,data element

Parameters

number	An integer from 0 to 29 that identifies the register to use.
data element	(1) The name of a single column (2) The content of a register (R*n*) (3) A number enclosed in the quotation marks (a numeric literal). The quotation marks must be specified, but DBA/Query will strip them and use only the number in the operation.
LOAD	Replaces the current contents of the register with the <data element>.
ADD	Adds the <data element> to the contents of the register.
SUBTRACT	Subtracts the <data element> from the contents of the register.
MULTIPLY	Multiplies the contents of the register by the <data element>.
DIVIDE	Divides the contents of the register by the <data element>.

Discussion

OPERATION	After each operation is executed, the result is placed in the register specified at the beginning

of the statement. For example, assuming R2 contains 1 and R4 contains 5, after the statement (R2,MULTIPLY,R4) is executed, R2 will contain 6 and R4 remains the same.

INITIAL VALUE Each register is initialized to zero when the REPORT begins execution.

COLUMN TYPE Only numeric type column can be used in register operations. They are DECIMAL, NUMERIC, FLOAT, DOUBLE PRECISION, REAL, INTEGER and SMALLINT.

EXAMPLE

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> B=POLICY.TESTDB.LEETECH
(2)>> S=DBA.CLAIM
(3)>> F POLICY='POL0001'
5 entries qualified
(4)>> R
RW>> H1,"DATE:",5
RW>> H1,DATE,15
RW>> H1,"CLAIMS FOR POLICY - POL0001",60
RW>> H3,"CLAIM NO",9
RW>> H3,"TOTAL AMOUNT",25
RW>> H3,"PAID AMOUNT",40
RW>> H3,"==========",10
RW>> H4,"==============",26
RW>> H4,"==============",42
RW>> E1,"$$$,$$$,$$$.99"
RW>> D1,CLAIM_NO,10
RW>> D1,TOTAL_AMOUNT,26,E1
RW>> D1,PAID_AMOUNT,42,E1
RW>> R1,ADD,TOTAL_AMOUNT
RW>> R2,ADD,PAID_AMOUNT
RW>> TF,"TOTAL",10
RW>> TF,R1,26,E1
RW>> TF,R2,42,E1
RW>> END
DATE:07/24/1991   CLAIMS FOR POLICY - POL0001

CLAIM NO    TOTAL AMOUNT   PAID AMOUNT
----------  ------------   --------------
CLM0000001       $900.00             $.00
CLM0000005     $1,000.00          $250.00
CLM0000004     $8,000.00           $50.00
CLM0000002     $1,200.00        $1,200.00
CLM0000003     $1,000.00             $.00
TOTAL         $12,100.00        $1,500.00
```

REPORT - Sort

S*level*, data item name [,{ASC|DES}]

Parameters

level — An integer anywhere from 1 to 10. A sort statement with level greater than 1 must be accompanied by those sort statements containing all lower level numbers. That is, if S3 appears in the report, S2 and S1 must also appear in it. There is no need for specification of a level. In this case, the sort statement sorts but does not define a control break.

data item name — The name of a single column.

ASC — Indicates that the data item values are to be ordered in ascending order. If ASC or DES is not specified, then the default order is ASC.

DES — Indicates that the data item values are to be ordered in descending order.

Discussion

CONTROL BREAKS — A control break occurs during the printing of a report whenever the value of a current entry for a data item defined in a numbered sort statement is different from the value of the last entry. When the first entry is printed, a control break occurs because the data item value changes from no value to the first value. Totals are not printed when the first control break occurs.

MAJOR TO MINOR — Numbered and unnumbered sort statements appear in the same REPORT request. The order in which sort statements appear in the REPORT BODY is significant.

```
1) MAJOR -----------------------------> MINOR
```

```
   S10  S9  S8  S7  S6  S5  S4  S3  S2  S1

2) RW>>S1,POLICY_NO
   RW>>S1,CLAIM_NO
   RW>>S1,FILE_DATE

   MAJOR ------------------------------> MINOR
   POLICY_NO        CLAIM_NO        FILE_DATE
```

MAXIMUM SORT ITEMS The Maximum number of sort statements allowed in a single report is 66.

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001   California Compensation
(1)>> B=POLICY.TESTDB.LEETECH
(2)>> S=DBA.CLAIM
(3)>> F POLICY='POL0001'
5 entries qualified
(4)>> R
RW>> H1,"DATE:",5
RW>> H1,DATE,15
RW>> H1,"CLAIMS FOR POLICY - POL0001",60
RW>> H3,"CLIENT NO",9
RW>> H3,"CLAIM NO",23
RW>> H4,"==========",10
RW>> H4,"==========",25
RW>> S1,CLIENT_NO
RW>> S1,CLAIM_NO
RW>> D1,CLIENT_NO,10
RW>>D1,CLAIM_NO,25
RW>>END

DATE:07/24/1991   CLAIMS FOR POLICY - POL0001

CLIENT NO         CLAIM NO
--------------------      --------------------
CLT0000001        CLM0000001
CLT0000001        CLM0000005
CLT0000006        CLM0000004
```

```
      CLT0000008        CLM0000002
      CLT0000008        CLM0000003
```

REPORT - Total

T*level*, print element, print position [,SPACE A[*number*]]
[,SPACE B[*number*]][,SKIP{A|B}][,E{*number*|Z}]
[{,ADD|,AVERAGE|,COUNT}]

or the special form:

T*level*, R*n*

Parameters

level

The letter F or an integer anywhere from 1 to 10, corresponding to the level of a sort statement. The letter F indicates that the information is to be printed only at the end of the report, after the last detail line, and it is related to the entire report, not just a subgroup.

print element

(1) The name of a single column. The value of the specified data column is printed.

(2) The content of a register (R*n*).

(3) A number enclosed in the quotation marks (a numeric literal). The quotation marks must be specified, but DBA/Query will strip them when it prints the characters.

print position

Refer to Table 5-2 Summary of Report Writer Command Parameters.

SPACE A*number*
SPACE B*number*
SKIP {A|B}
E{*number*|Z}

ADD

Prints the control group total values for the data item specified as <print element>. The <print element> must be the name of a numeric type data item.

AVERAGE — Prints the control group average value for the data item specified as the <print element>. The <print element> must be the name of a numeric type data item.

COUNT — Prints a count of the number of values for that control group. The <print element> must be the name of a data item.

R*n* — The register to be cleared

Discussion

Each control break occurs as a result of a sort statement labeled from 1 to 10. When the control break occurs, the total statement corresponding to the sort level, which causes the break, prints the information specified. Total statements that are NOT labeled TF require the corresponding sort statements.

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> B=POLICY.TESTDB.LEETECH
(2)>> S=DBA.CLAIM
(3)>> F POLICY='POL0001'
5 entries qualified
(4)>> R
RW>>H1,"DATE:",5
RW>>H1,DATE,15
RW>>H1,"CLAIMS FOR POLICY - POL0001",60
RW>>H3,"CLIENT NO",9
RW>>H3,"CLAIM NO",23
RW>>H4,"==========",10
RW>>H4,"==========",25
RW>>S1,CLIENT_NO
RW>>D1,CLIENT_NO,10
RW>>D1,CLAIM_NO,25
RW>>T1,"* CLIENT BREAK *",20
RW>>TF,"TOTAL NUMBER OF CLIENTS = ",25
RW>>TF,CLIENT_NO,27,COUNT
RW>>END


DATE:07/24/1991   CLAIMS FOR POLICY - POL0001

CLIENT NO         CLAIM NO
----------        ----------
CLT0000001        CLM0000001
CLT0000001        CLM0000005
      * CLIENT BREAK *
CLT0000006        CLM0000004
      * CLIENT BREAK *
CLT0000008        CLM0000002
CLT0000008        CLM0000003
      * CLIENT BREAK *
TOTAL NUMBER OF CLIENTS = 5
```

SET COLTYPE

SET COLTYPE { ON | OFF }

Parameters

ON	Turns on the COLUMN TYPE DISPLAY for row addition.
OFF	Turns off the COLUMN TYPE DISPLAY for row addition, and the default is OFF.

Discussion

In row addition function (ADD), the column name is prompted for user input. If the COLTYPE is ON, then the column type is also displayed along with the column name.

Example

Add a row to table DBA.CLAIM.

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> B=POLICY.TESTDB.LEETECH
(2)>> S=DBA.CLAIM
(3)>> ADD
CLAIM_NO             =>>CLM0000088
CLAIMANT_NO          =>>
...
...
(4)>> SET COLTYPE ON
(5)>> ADD
CLAIM_NO       CHAR(10)    =>>CLM0000088
CLAIMANT_NO    CHAR(10)    =>>
...
...
(3)>>
```

SET DOUBLEQUOTE

SET DOUBLEQUOTE { ON | OFF }

Parameters

ON — Turns on SET DOUBLEQUOTE option.

OFF — Turns off SET DOUBLEQUOTE option, and default is OFF.

Discussion

When SET DOUBLEQUOTE ON, DBA/Query treats the doublequote “” the same as the singlequote ‘’ . The default is OFF.

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> B=POLICY.TESTDB.LEETECH
(2)>> S=DBA.CLAIM
(3)>> F CLAIM_NO = "CLM0000001"
0 entries qualified
SQL ERROR (DBAQERR -5003)
Column CLM0000001 not found (DBERR 2211)
(4)>> SET DOUBLEQUOTE ON
(5)>> F CLAIM_NO = "CLM0000001"
1 entries qualified
```

SET ERRDIAG

SET ERRDIAG { ON | OFF }

Parameters

ON Turns on COMMAND SYNTAX CHECKER.

OFF Turns off COMMAND SYNTAX CHECKER (default).

Discussion

The command syntax checker diagnoses command error and displays appropriate help messages.

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001     California Compensation
(1)>> B=POLICY.TESTDB.LEETECH
(2)>> S=DBA.CLAIM
(3)>> FF
0 entries qualified
SYNTAX ERROR (DBAQERR -3006)
(4)>> SET ERRDIAG ON
(5)>> FF
0 entries qualified

SYNTAX - 1)   F       [#LIMIT=i]
              F ALL   [#LIMIT=i]
              FIND ALL[#LIMIT=i]

         2)   F    [#LIMIT=i] {selection criteria}
              FIND [#LIMIT=i] {selection criteria}

         3)   FO|FORM [COLUMN] [[owner.]table|view]
              FO|FORM LOCK
              FO|FORM RUNTIME
              FO|FORM SET[S]   [DBEFileSet]
                               [,lower,upper[,SCAN]]
```

```
             FO|FORM TABLE    [[owner.]table|view]
                              [,COUNT]
             FO|FORM TABLE #Snnn|#Jnnn
             FO|FORM USER  [session,][user.account]

SYNTAX ERROR (DBAQERR -3006)
(6)>>
```

SET HYPHEN

SET HYPHEN { ON | OFF }

Parameters

ON Turns on SET HYPHEN option.

OFF Turns off SET HYPHEN option, and default is OFF.

Discussion

When SET HYPHEN ON, DBA/QUERY treats the hyphen '-' the same as the underscore '_'. The default is OFF.

Example

```
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> B=POLICY.TESTDB.LEETECH
(2)>> S=DBA.CLAIM
(3)>> F CLAIM-NO = 'CLM0000001'
0 entries qualified
SQL ERROR (DBAQERR -5003)
Column CLAIM not found (DBERR 2211)
Column NO not found (DBERR 2211)
(4)>> SET HYPHEN ON
(5)>> F CLAIM-NO = 'CLM0000001'
1 entries qualified
```

SHOW

SHOW

Discussion

This command shows current database, active table or view, output device, and control variables values. The database's actual name and user input name are both displayed to avoid the confusion caused by MPE/XL FILE equation command.

Example

```
:FILE DBE=POLICY.TESTDB.LEETECH
:RUN DBAQ.PUB.LEETECH

DBA/QUERY(A.00.01) LeeTech Software Inc. Copyright
1992.
License No. 000001       California Compensation
(1)>> B=DBE
(2)>> S=DBA.CLAIM
(3)>> SHOW
DATABASE   = DBE (POLICY.TESTDB.LEETECH)
TABLE/VIEW = DBA.CLAIM
OUT        = TERM
COLTYPE    = OFF
ERRDIAG    = OFF
DOUBLEQUOTE= OFF
HYPHEN     = OFF
(4)>>
```

6. Control Commands

DBA/QUERY Control Command Summary:

COMMENT or # {text} Any text may be put after the comment command and DBA/Query will totally ignore it.

EXIT or // Ends the DBA/Query program.

HELP or HELP {command} Invokes the DBA/Query on-line help facility. HELP gives a summary level help, and the HELP {command} gives specific command level help information.

LISTREDO Displays DBA/Query command history.

REDO Redo and modify a previous DBA/Query command.

REDO Redoes the previous command.

REDO {n} Redoes the command with the command number n in DBA/Query history.

REDO {-n} Redoes the nth command before the most recent one (-1).

REDO {string} Redoes the most recent command beginning with the string.

Modify Commands:

i	insert
r	replace
d	delete

Any other characters - implied replace

DO Does a previous DBA/QUERY command.

X or XEQ {script file} Executes a set of DBA/QUERY commands from a standard ASCII and unnumbered file.

A. Sample POLICY Database Schema

Database Configuration and Log Files

```
START DBE 'POLICY' MULTI NEW
      BUFFER      = ( 50, 50 ),
      TRANSACTION = 10,
      DBEFILE0 DBEFILE INITIAL_DBEFILE
          WITH PAGES = 100,
            NAME  = 'POLICY00',
      LOG DBEFILE POLICY_LOG
          WITH PAGES = 256,
            NAME  = 'POLLOG01';
```

DBEFileSets and Physical Database Storage Allocation

```
CONNECT TO 'POLICY';


CREATE DBEFILESET NEW_POLICY_SET;

CREATE DBEFILESET OLD_POLICY_SET;

CREATE DBEFILESET CLAIM_SET;

CREATE DBEFILESET PAYMENT_SET;

CREATE DBEFILESET XREF_TABLE_SET;


CREATE DBEFILE POLICY_DATA_001
WITH PAGES=10,
NAME='POLICY01',
TYPE=TABLE;

CREATE DBEFILE POLICY_DATA_002
WITH PAGES=10,
NAME='POLICY02',
TYPE=TABLE;

CREATE DBEFILE POLICY_INDEX_01
WITH PAGES=10,
NAME='POLICY03',
TYPE=INDEX;

CREATE DBEFILE OLDPOL_DATA_001
WITH PAGES=10,
NAME='OLDPOL01',
TYPE=TABLE;

CREATE DBEFILE OLDPOL_DATA_002
WITH PAGES=10,
NAME='OLDPOL02',
TYPE=MIXED;

CREATE DBEFILE OLDPOL_INDEX_01
WITH PAGES=10,
NAME='OLDPOL03',
TYPE=INDEX;

CREATE DBEFILE CLAIM_DATA_001
WITH PAGES=10,
NAME='CLAIM001',
TYPE=TABLE;

CREATE DBEFILE CLAIM_INDEX_01
WITH PAGES=10,
NAME='CLAIM002',
TYPE=INDEX;

CREATE DBEFILE PAY_DATA_001
WITH PAGES=10,
NAME='PAYMNT01',
TYPE=TABLE;

CREATE DBEFILE PAY_INDEX_01
WITH PAGES=10,
NAME='PAYMNT02',
TYPE=INDEX;

CREATE DBEFILE XREF_TABLE_001
WITH PAGES=10,
NAME='XTABLE01',
TYPE=MIXED;


ADD DBEFILE POLICY_DATA_001
TO DBEFILESET NEW_POLICY_SET;

ADD DBEFILE POLICY_DATA_002
TO DBEFILESET NEW_POLICY_SET;

ADD DBEFILE POLICY_INDEX_01
TO DBEFILESET NEW_POLICY_SET;

ADD DBEFILE OLDPOL_DATA_001
TO DBEFILESET OLD_POLICY_SET;

ADD DBEFILE OLDPOL_DATA_002
TO DBEFILESET OLD_POLICY_SET;

ADD DBEFILE OLDPOL_INDEX_01
TO DBEFILESET OLD_POLICY_SET;

ADD DBEFILE CLAIM_DATA_001
TO DBEFILESET CLAIM_SET;

ADD DBEFILE CLAIM_INDEX_01
TO DBEFILESET CLAIM_SET;

ADD DBEFILE PAY_DATA_001
TO DBEFILESET PAYMENT_SET;

ADD DBEFILE PAY_INDEX_01
TO DBEFILESET PAYMENT_SET;
```

```
ADD DBEFILE XREF_TABLE_001
TO DBEFILESET XREF_TABLE_SET;

COMMIT WORK;
```

Logical Database Tables

```
CONNECT TO 'POLICY';


CREATE PUBLIC TABLE DBA.ACTIVE_POLICY
    ( POLICY_NO         CHAR(10)    NOT NULL,
      CLIENT_NO         CHAR(10)    NOT NULL,
      POLICY_BEGIN_DATE INTEGER     NOT NULL,
      POLICY_END_DATE   INTEGER     NOT NULL,
      POLICY_TERM       CHAR(80)    )
IN NEW_POLICY_SET;


CREATE PUBLIC TABLE DBA.INACTIVE_POLICY
    ( POLICY_NO         CHAR(10)    NOT NULL,
      CLIENT_NO         CHAR(10)    NOT NULL,
      POLICY_BEGIN_DATE INTEGER     NOT NULL,
      POLICY_END_DATE   INTEGER     NOT NULL,
      POLICY_TERM       CHAR(80)            ,
      ARCHIVE_DATE      INTEGER     NOT NULL  )
IN OLD_POLICY_SET;


CREATE PUBLIC TABLE DBA.CLAIM
    ( CLAIM_NO          CHAR(10)    NOT NULL,
      CLAIMANT_NO       CHAR(10)    NOT NULL,
      POLICY_NO         CHAR(10)    NOT NULL,
      CLIENT_NO         CHAR(10)    NOT NULL,
      FILE_DATE         INTEGER     NOT NULL,
      TYPE_OF_INJURY    CHAR(02)    NOT NULL,
      TOTAL_AMOUNT      INTEGER     NOT NULL,
      PAID_AMOUNT       INTEGER              )
UNIQUE HASH ON ( CLAIM_NO ) PAGES = 6 IN CLAIM_SET;

CREATE PUBLIC TABLE DBA.PAYMENT
    ( CLAIM_NO          CHAR(10)    NOT NULL,
      CHECK_NO          CHAR(20)    NOT NULL,
      CHECK_AMOUNT      INTEGER     NOT NULL,
      CHECK_DATE        INTEGER     NOT NULL  )
```

```
IN PAYMENT_SET;


CREATE PUBLIC TABLE DBA.CLAIMANT
    ( CLAIMANT_NO       CHAR(10)    NOT NULL,
      CLAIMANT_NAME     CHAR(40)    NOT NULL,
      CLAIMANT_ADDRESS  CHAR(120)         )
IN XREF_TABLE_SET;


CREATE PUBLIC TABLE DBA.CLIENT
    ( CLIENT_NO         CHAR(10)    NOT NULL,
      CLIENT_NAME       CHAR(40)    NOT NULL,
      CLIENT_ADDRESS    CHAR(120)         )
IN XREF_TABLE_SET;


CREATE CLUSTERING
INDEX DBA.ACTIVE_POLICY_IDX
ON DBA.ACTIVE_POLICY
( POLICY_NO );

CREATE
INDEX DBA.ACTIVE_CLIENT_IDX
ON DBA.ACTIVE_POLICY
( CLIENT_NO );

CREATE CLUSTERING
INDEX DBA.INACTIVE_POLICY_IDX
ON DBA.INACTIVE_POLICY
( POLICY_NO );

CREATE UNIQUE
INDEX DBA.INACTIVE_CLIENT_IDX
```

```
ON DBA.INACTIVE_POLICY
(CLIENT_NO, POLICY_NO );

CREATE
INDEX DBA.CLAIM_IDX
ON DBA.CLAIM
( CLAIMANT_NO );

CREATE
INDEX DBA.PAYMENT_IDX
ON DBA.PAYMENT
( CLAIM_NO );
CREATE
INDEX DBA.CHECK_IDX
ON DBA.PAYMENT
( CHECK_NO );

CREATE UNIQUE
INDEX DBA.CLAIMANT_IDX
ON DBA.CLAIMA
( CLAIMANT_NO );

CREATE UNIQUE
INDEX DBA.CLIENT_IDX
ON DBA.CLIENT
( CLIENT_NO );


COMMIT WORK;
```

Database Views

```
CONNECT TO 'POLICY';


CREATE VIEW DBA.ACTIVE_POLICY_CLIENT
    (  POLICY_NO,
      CLIENT_NO,
      CLIENT_NAME,
      CLIENT_ADDRESS,
      POLICY_BEGIN_DATE,
      POLICY_END_DATE,
      POLICY_TERM       ) AS

SELECT DBA.ACTIVE_POLICY.POLICY_NO,
      DBA.ACTIVE_POLICY.CLIENT_NO,
      DBA.CLIENT.CLIENT_NAME,
      DBA.CLIENT.CLIENT_ADDRESS,
      DBA.ACTIVE_POLICY.POLICY_BEGIN_DATE,
      DBA.ACTIVE_POLICY.POLICY_END_DATE,
      DBA.ACTIVE_POLICY.POLICY_TERM

FROM DBA.ACTIVE_POLICY,
      DBA.CLIENT

WHERE DBA.ACTIVE_POLICY.CLIENT_NO =
      DBA.CLIENT.CLIENT_NO;


CREATE VIEW DBA.CLAIMANT_PAYMENT
    ( CLAIM_NO,
      TOTAL_AMOUNT,
      PAID_AMOUNT,
      CLAIMANT_NO,
      CLAIMANT_NAME,
      CLAIMANT_ADDRESS,
      CHECK_NO,
      CHECK_AMOUNT,
      CHECK_DATE        ) AS

SELECT DBA.CLAIM.CLAIM_NO,
      DBA.CLAIM.TOTAL_AMOUNT,
      DBA.CLAIM.PAID_AMOUNT,
      DBA.CLAIM.CLAIMANT_NO,
      DBA.CLAIMANT.CLAIMANT_NAME,
      DBA.CLAIMANT.CLAIMANT_ADDRESS,
      DBA.PAYMENT.CHECK_NO,
      DBA.PAYMENT.CHECK_AMOUNT,
      DBA.PAYMENT.CHECK_DATE
```

```
FROM DBA.CLAIM,
            DBA.CLAIMANT,
            DBA.PAYMENT

      WHERE DBA.CLAIM.CLAIM_NO
            = DBA.PAYMENT.CLAIM_NO
            AND DBA.CLAIM.CLAIMANT_NO
            = DBA.CLAIMANT.CLAIMANT_NO;


      COMMIT_WORK;
```

Database Security and Authority

```
      CONNECT TO 'POLICY';


      CREATE GROUP DBA.POLICY_UNDERWRITER;

      CREATE GROUP DBA.CLAIM_DEPARTMENT;

      CREATE GROUP DBA.ACCOUNTS_PAYABLE;

      CREATE GROUP DBA.MIS_DEVELOPMENT;

      CREATE GROUP DBA.MIS_MAINTENANCE;

      CREATE GROUP DBA.MIS_OPERATION;


      GRANT SELECT, INSERT, DELETE, UPDATE
      ON DBA.ACTIVE_POLICY
      TO POLICY_UNDERWRITER;

      GRANT SELECT, INSERT, DELETE, UPDATE
      ON DBA.INACTIVE_POLICY
      TO POLICY_UNDERWRITER;

      GRANT SELECT, INSERT, DELETE, UPDATE
      ON DBA.CLIENT
      TO POLICY_UNDERWRITER;

      GRANT SELECT
      ON DBA.ACTIVE_POLICY_CLIENT
      TO POLICY_UNDERWRITER;
```

```
GRANT SELECT, INSERT, DELETE, UPDATE
ON DBA.CLAIM
TO CLAIM_DEPARTMENT;

GRANT SELECT, INSERT, DELETE, UPDATE
ON DBA.CLAIMANT
TO CLAIM_DEPARTMENT;

GRANT SELECT
ON DBA.ACTIVE_POLICY
TO CLAIM_DEPARTMENT;

GRANT SELECT
ON DBA.CLAIMANT_PAYMENT
TO CLAIM_DEPARTMENT;


GRANT SELECT, INSERT, DELETE, UPDATE
ON DBA.PAYMENT
TO ACCOUNTS_PAYABLE;

GRANT SELECT
ON DBA.CLAIMANT_PAYMENT
TO ACCOUNTS_PAYABLE;

GRANT SELECT
ON DBA.ACTIVE_POLICY
TO ACCOUNTS_PAYABLE;

GRANT SELECT
ON DBA.CLAIM
TO ACCOUNTS_PAYABLE;
GRANT SELECT
ON DBA.CLAIMANT
TO ACCOUNTS_PAYABLE;


GRANT SELECT, ALTER, INDEX
ON DBA.ACTIVE_POLICY
TO MIS_MAINTENANCE;

GRANT SELECT, ALTER, INDEX
ON DBA.INACTIVE_POLICY
```

```
TO MIS_MAINTENANCE;

GRANT SELECT, ALTER, INDEX
ON DBA.CLAIM
TO MIS_MAINTENANCE;

GRANT SELECT, ALTER, INDEX
ON DBA.PAYMENT
TO MIS_MAINTENANCE;

GRANT SELECT, ALTER, INDEX
ON DBA.CLIENT
TO MIS_MAINTENANCE;

GRANT SELECT, ALTER, INDEX
ON DBA.CLAIMANT
TO MIS_MAINTENANCE;


GRANT SELECT, INDEX
ON DBA.ACTIVE_POLICY
TO MIS_OPERATION;

GRANT SELECT, INDEX
ON DBA.INACTIVE_POLICY
TO MIS_OPERATION;

GRANT SELECT, INDEX
ON DBA.CLAIM
TO MIS_OPERATION;

GRANT SELECT, INDEX
ON DBA.PAYMENT
TO MIS_OPERATION;

GRANT SELECT, INDEX
ON DBA.CLIENT
TO MIS_OPERATION;

GRANT SELECT, INDEX
ON DBA.CLAIMANT
TO MIS_OPERATION;


GRANT CONNECT
```

```
TO POLICY_UNDERWRITER;

GRANT CONNECT
TO CLAIM_DEPARTMENT;

GRANT CONNECT
TO ACCOUNTS_PAYABLE;

GRANT DBA
TO MIS_DEVELOPMENT;

GRANT CONNECT
TO MIS_MAINTENANCE;
GRANT RESOURCE
TO MIS_OPERATION;


ADD   RON@POLICY,
      PETER@POLICY,
      NANCY@POLICY,
      RAY@POLICY
TO GROUP POLICY_UNDERWRITER;
```

```
ADD   MARY@POLICY,
      SKIP@POLICY,
      DANIEL@POLICY
TO GROUP CLAIM_DEPARTMENT;


ADD   LIBBY@POLICY,
      MYRA@POLICY,
      DANA@POLICY
TO GROUP ACCOUNTS_PAYABLE;


ADD   BOB@POLICY,
      OLGA@POLICY
TO GROUP MIS_DEVELOPMENT;


ADD   SAM@POLICY,
      FRANK@POLICY
TO GROUP MIS_MAINTENANCE;


ADD   DAVID@POLICY,
      VICTOR@POLICY,
      LEE@POLICY
TO GROUP MIS_OPERATION;


COMMIT WORK;
```

POLICY Database Overview

POLICY DBEnvironment is a collection of 5 logical databases (in HP/SQL term), NEW_POLICY_SET, OLD_POLICY_SET, CLAIM_SET, PAYMENT_SET, and XREF_TABLE_SET.

The NEW_POLICY_SET database contains all active insurance policies which can be accessed either by clustering index POLICY_NO or b-tree index CLIENT_NO.

The OLD_POLICY_SET database contains all inactive insurance policies which can be accessed either by clustering index POLICY_NO or by a concatenation of b-tree index CLIENT_NO and POLICY_NO.

The CLAIM_SET database contains insurance claims, and each claim can be directly addressed via the hash index structure of CLAIM_NO.

The PAYMENT_SET database contains claim's payment history with b-tree index CLAIM_NO or CHECK_NO.

The XREF_TABLE_SET database contains miscellaneous cross reference tables, CLIENT for policy holders, and CLAIMANT.

There are two views which are defined across database boundary, and they are designed for quick reporting or on-line inquiry.

The database security and access authorization for POLICY are:

- POLICY UNDERWRITER can read and write active policy, inactive policy, and client’s information;
- CLAIM DEPARTMENT can read and write claim and claimant's information, but can only inquire active policies;
- ACCOUNTS PAYABLE can change claimant’s payments, but is limited to read only policy, claim, and claimant’s information;
- MIS DEVELOPMENT group has the DBA capability of a DBEnvironment;
- MIS MAINTENANCE group can only read data, append new columns to the end of each table, and drop or build table’s indexes;
- MIS OPERATION can look at the data and drop or build table’s indexes.

Mapping of MPE/XL Logons and POLICY Database Security Provisions

The logons, RON.POLICY, PETER.POLICY, NANCY.POLICY, and RAY.POLICY have the same database access authorities and security constrains as the group POLICY_UNDERWRITER. The same rules apply for: MARY.POLICY, SKIP.POLICY and DANIEL.POLICY to group CLAIM_DEPARTMENT; LIBBY.POLICY, MYRA.POLICY and DANA.POLICY to group ACCOUNTS_PAYABLE; BOB.POLICY and OLGA.POLICY to group MIS_DEPARTMENT; SAM.POLICY and

FRANK.POLICY to group MIS_MAINTENANCE; and DAVID.POLICY, VICTOR.POLICY and LEE.POLICY to group MIS_OPERATION. The MPE/XL logon is in the format of USER.ACCOUNT, but in HP/SQL database, the presentation of an MPE/XL user is USER@ACCOUNT. The '.' is replaced by '@' due to the conflicting usage of HP/SQL object's ownership qualification, such as, OWNER.TABLE, OWNER.VIEW, ... etc.

```
              Logical Database
DBEnvironment Name              Table Name          Index Type  Index Name
============= ================  =================   =========== ==================
POLICY        NEW_POLICY_SET    DBA.ACTIVE_POLICY   CLUSTERING  ACTIVE_POLICY_IDX
                                                    B-TREE      ACTIVE_CLIENT_IDX
              OLD_POLICY_SET    DBA.INACTIVE_POLICY CLUSTERING  INACTIVE_POLICY_IDX
                                                    B-TREE      INACTIVE_CLIENT_IDX
              CLAIM_SET         DBA.CLAIM           HASH
                                                    B-TREE      CLAIM_IDX
              PAYMENT_SET       DBA.PAYMENT         B-TREE      PAYMENT_IDX
                                                    B-TREE      CHECK_IDX
              XREF_TABLE_SET    DBA.CLAIMANT        B-TREE      CLAIMANT_IDX
                                DBA.CLIENT          B-TREE      CLIENT_IDX
```

Example A-1. POLICY Database Structure

```
Group Name          Authority  Member        Table Name                SEL INS DEL UPD ALT IDX
=============       ========   ============  ====================      === === === === === ===
POLICY_UNDERWRITER  CONNECT    RON.POLICY    DBA.ACTIVE_POLICY          Y   Y   Y   Y
                               PETER.POLICY  DBA.INACTIVE_POLICY        Y   Y   Y   Y
                               NAN.POLICY    DBA.CLIENT                 Y   Y   Y   Y
                               RAY.POLICY    DBA.ACTIVE_POLICY_CLIENT   Y
CLAIM_DEPARTMENT    CONNECT    MARY.POLICY   DBA.CLAIM                  Y   Y   Y   Y
                               SKIP.POLICY   DBA.CLAIMANT               Y   Y   Y   Y
                               DAN.POLICY    DBA.ACTIVE_POLICY          Y
                                             DBA.CLAIMANT_PAYMENT       Y
ACCOUNTS_PAYABLE    CONNECT    LIBBY.POLICY  DBA.PAYMENT                Y   Y   Y   Y
                               MYRA.POLICY   DBA.CLAIMANT_PAYMENT       Y
                               DANA.POLICY   DBA.ACTIVE_POLICY          Y
                                             DBA.CLAIM                  Y
                                             DBA.CLAIMANT               Y
MIS_DEVELOPMENT     DBA        BOB.POLICY
                               OLGA.POLICY
MIS_MAINTENANCE     CONNECT    SAM.POLICY    DBA.ACTIVE_POLICY          Y               Y   Y
                               FRAN.POLICY   DBA.INACTIVE_POLICY        Y               Y   Y
                                             DBA.CLAIM                  Y               Y   Y
                                             DBA.PAYMENT                Y               Y   Y
                                             DBA.CLIENT                 Y               Y   Y
                                             DBA.CLAIMANT               Y               Y   Y
MIS_OPERATION       RESOURCE   DAVE.POLICY   DBA.ACTIVE_POLICY          Y               Y   Y
                               VIC.POLICY    DBA.INACTIVE_POLICY        Y               Y   Y
                               LEE.POLICY    DBA.CLAIM                  Y               Y   Y
                                             DBA.PAYMENT                Y               Y   Y
                                             DBA.CLIENT                 Y               Y   Y
                                             DBA.CLAIMANT               Y               Y   Y
```

Example A-2 POLICY Databse Securit Provisions

B. Error Messages

-2001 ~ -2010

MESSAGE **DBAQ ERROR**

CAUSE A file system error occurred while DBA/Query was manipulating files.

ACTION Contact a system administrator.

-3001 MESSAGE **INVALID COMMAND**

CAUSE DBA/Query can not recognize the command.

ACTION Refer to the command description for the proper format and re-enter the command.

-3002 MESSAGE **SYNTAX ERROR**

CAUSE The syntax is incorrect.

ACTION Refer to the DBA/Query command description for correct syntax and re-enter the command.

-3002 MESSAGE **EXPECTED ' ='**

CAUSE A equal sign is missing from the command.

ACTION Correct the command or statement and re-enter it.

-3002 MESSAGE **OPTION ERROR**

CAUSE The command uses an invalid option.

ACTION Correct the command or statement and re-enter it.

-3002	MESSAGE	**EXPECTED ' TO'**
	CAUSE	A TO number is missing from the data.
	ACTION	Correct the command or statement and re-enter it.
-3003	MESSAGE	**SYNTAX ERROR**
	CAUSE	The syntax is incorrect.
	ACTION	Refer to the DBA/Query command description for correct syntax and re-enter the command.
-3002	MESSAGE	**Table/View REQUIRED**
	CAUSE	A table/view name after the command is missing.
	ACTION	Re-enter the command and the required table or view.
-3003	MESSAGE	**DBEname REQUIRED**
	CAUSE	A DBEnvironment is missing after the command.
	ACTION	Re-enter the command and the required DBEnvironment.
-3004	MESSAGE	**BUFFER OVERFLOW**
	CAUSE	The total length of a data item name exceeds the predefined size.
	ACTION	Rename the item or contact DBA/Query support.

-3005	MESSAGE	**NUMBER REQUIRED**
	CAUSE	A number is missing from the command.
	ACTION	Re-enter the command and a number required.
-3006	MESSAGE	**SYNTAX ERROR**
	CAUSE	The syntax is incorrect.
	ACTION	Refer to the DBA/Query command description for correct syntax and re-enter the command.
-3006	MESSAGE	**THE @ SIGN IS NOT ALLOWED IN BOTH SIDES OF THE 'TO'**
	CAUSE	The @ sign appears at the both side of 'TO' in a column equivalence.
	ACTION	Correct the command and re-enter it.
-3006	MESSAGE	**EXPECTED SPACE**
	CAUSE	A space is missing in the command.
	ACTION	Refer to the DBA/Query command description and re-enter it.
-4001	MESSAGE	**DBAQ ERROR**
	CAUSE	This is an internal DBA/Query problem.
	ACTION	Contact DBA/Query support.

-4002	MESSAGE	**DBEname TOO LONG**
	CAUSE	The name of the DBEnvironment is too long.
	ACTION	Reduce the length of DBEnvironment and re-enter it.
-4003	MESSAGE	**DBAQ ERROR**
	CAUSE	This is an internal DBA/Query problem.
	ACTION	Contact DBA/Query support.
-4004	MESSAGE	**TABLE NAME TOO LONG**
	CAUSE	The length of table or view is too long.
	ACTION	Reduce the length and re-enter the command.
-4005	MESSAGE	**FILE NAME TOO LONG**
	CAUSE	The length of the file name is too long.
	ACTION	Reduce the length of the file name and re-enter the command.
-4006	MESSAGE	**RECORD HAS NOT YET BEEN FOUND**
	CAUSE	The command has been executed prior to selecting data by retrieval command.
	ACTION	First use the retrieval command, such as FIND, then re-enter this command.

-4007	MESSAGE	**DBAQ ERROR**
	CAUSE	This is an internal DBA/Query problem.
	ACTION	Contact DBA/Query support.
-4008	MESSAGE	**INVALID NUMBER**
	CAUSE	The number after #LIMIT= is invalid.
	ACTION	Re-enter a positive integer number for the command.
-4009	MESSAGE	**ILLEGAL NAME**
	CAUSE	The length of a COLUMN name is too long or the syntax of it is illegal.
	ACTION	Re-enter a correct column name for the command.
-4010	MESSAGE	**EXPECTED A ' .'**
	CAUSE	A '.' is missing for a column name in the column equivalence used in JOIN command.
	ACTION	Re-enter the correct column name.
-5001	MESSAGE	**DATABASE CAN NOT BE OPENED**
	CAUSE	Can not open the DBEnvironment.
	ACTION	Check the connection authority or the location of the DBEnvironment.

-5001	MESSAGE	**NO SUCH DATABASE**
	CAUSE	No such database exists.
	ACTION	Check the spelling or the location of the DBEnvironment.
-5002	MESSAGE	**SQL SYNTAX ERROR**
	CAUSE	Incorrect SQL syntax is used in selection criteria.
	ACTION	Re-enter a correct syntax expression for the command.
-5003	MESSAGE	**SQL ERROR**
	CAUSE	A SQL error happened.
	ACTION	Check the SQL error message following the DBAQ error message. If this message persists, consult with a database administrator.
-5003	MESSAGE	**Table/View NOT FOUND**
	CAUSE	Can not find the table or view.
	ACTION	Check the spelling or location of the table or view.
-5004 ~ -5008		
	MESSAGE	**SQL ERROR**
	CAUSE	A SQL error happened.
	ACTION	Check the SQL error message following the DBAQ error message. If this message persists, consult with a database administrator.
-5009	MESSAGE	**DATABASE NOT DECLARED**

	CAUSE	The database has not been declared yet.
	ACTION	Declare the database with DATA-BASE= command.
-5010 ~ -5012	MESSAGE	**SQL ERROR**
	CAUSE	A SQL error happened.
	ACTION	Check the SQL error message following the DBAQ error message. If this message persists, consult with a database administrator.
-5013	MESSAGE	**Table/View NOT DECLARED**
	CAUSE	The table or view has not been declared yet.
	ACTION	Declare the table or view with DATA-SETS= command.
-5014	MESSAGE	**DELETE FAIL**
	CAUSE	Deleting rows fails.
	ACTION	Re-try it. If this message persists, consult with a database administrator.
-5015	MESSAGE	**UPDATE FAIL**
	CAUSE	Updating rows fails.
	ACTION	Re-try it. If this message persists, consult with a database administrator.

-5016	MESSAGE	**COLUMN EXCEED MAXIMUM SIZE**
	CAUSE	Too many columns have been referenced in a JOIN statement.
	ACTION	Reduce the number of columns in a JOIN statement.
-5017	MESSAGE	**NO JOIN FILE EXISTS**
	CAUSE	The join statements have not been declared yet.
	ACTION	Declare the join statements with JOIN command.
-6001	MESSAGE	**OPEN INFO FILE FAIL**
	CAUSE	Information file has not been created or file system error.
	ACTION	Check the sequence of commands. If this message persists, contact DBA/Query support.
-6002	MESSAGE	**OPEN DATA FILE FAIL**
	CAUSE	Data file has not been created or file system error.
	ACTION	Check the sequence of commands. If this message persists, contact DBA/Query support.

C. Customer Support

LeeTech Software Inc. is dedicated to providing the highest quality software products available. In addition, LeeTech believes that high quality, knowledgeable support is an essential element for full utilization of the supplied products. LeeTech has thus structured a complete support package that is available with all supplied products.

All support contracts include telephone or message based support. LeeTech's staff is qualified to assist not only with product use, but also with any database problems customers might have. If problems prove extensive and outside the scope of the maintenance contract, LeeTech can still assist with low cost consulting.

In addition to specific service requests, LeeTech provides users with periodic newsletters, software bulletins and other materials designed to help ensure effective use of LeeTech products. If you with to be placed on the mailing list simply send your request to LeeTech.

LeeTech customer service can be contacted by any of the following.

Mail

Please send all inquiries or problems to:

LeeTech Software Inc.
2085 Hamilton Ave., Suite 200
San Jose, CA 95125
U.S.A.

Telephone Support

All LeeTech customers with a current support services contract can contact LeeTech at the following numbers.

U.S.A.	(800) 995-1987
All Other Countries	(408) 253-1987

Fax-Back Service

LeeTech maintains a 24-hour Fax number for customer input. Questions regarding support services receive immediate attention at the beginning of the following business day. Responses are returned by Fax or by the method designated on the submittal.

Fax Number (408) 558-0802

Electronic Messaging

Questions or input can be made via electronic mail.

- Support needs can be mailed to:

support@leetech.com

- Information inquiries can be mailed to:

info@leetech.com

- All product and sales needs can be mailed to:

sales@leetech.com

INDEX